Electric Drives and Controls | Hydraulics | Linear Motion and Assembly Technologies | Pneumatics | Service

Rexroth VisualMotion 11 Multi-Axis Machine Control Volume 2

R911309162
Edition 01

Functional Description

Title Rexroth VisualMotion 10
Multi-Axis Machine Control
Volume 2

Type of Documentation Functional Description

Document Typecode DOK-VISMOT-VM*-11VRS**-FK01-EN-P

Internal File Reference Document Number, 120-2300-B326-01/EN
Part of Box Set, 20-11V-EN (MN R911309160)

Purpose of Documentation This documentation describes the functionality supported in Rexroth VisualMotion 11 using G*P 11 firmware.

Record of Revisions

Description	Release Date	Notes
DOK-VISMOT-VM*-11VRS**-FK01-EN-P	4/2006	Offical release

Published by Bosch Rexroth AG
Bgm.-Dr.-Nebel-Str. 2 • D-97816 Lohr a. Main
Tel.: +49 (0)93 52/40-0 • Fax: +49 (0)93 52/40-48 85 • Telex: 68 94 21

Bosch Rexroth Corporation • Electric Drives and Controls
5150 Prairie Stone Parkway • Hoffman Estates, IL 60192 • USA
Tel.: 847-645-3600 • Fax: 847-645-6201

http://www.boschrexroth.com/

Dept. ESG4 (DPJ)

Note This document has been printed on chlorine-free bleached paper.

Table of Contents

12 About this Manual

12.1 Introduction

The VisualMotion Functional Description manual describes VisualMotion software from its components to how it used for basic programming. The manual is a two volume set. Volume 1 includes information on how to use VisualMotion to create a program. Volume 2 is designed as a reference guide which presents detailed descriptions of the features of the software.

12.2 Contents of Volume 2

The following table provides a description for the chapters available in volume 2:

Chapter	Description
Chapter 13	**VisualMotion Menus** Describes the menus in VisualMotion
Chapter 14	**VisualMotion Icons** Describes icon functionality in VisualMotion
Chapter 15	**Parameters** Describes control, task and axis parameters in VisualMotion
Chapter 16	**Registers** Describes the registers in VisualMotion
Chapter 17	**Compiler Warnings and Errors** Describes the compiler warnings and errors issued by VisualMotion
Chapter 18	**Communication Servers** Describes the DDE and OPC (SPC3) servers used with VisualMotion
Chapter 19	**Communication Protocols** Describes the communication protocols used with VisualMotion, including ASCII and SIS
Chapter 20	**PPC-P11.1 PCI Bus Interface** Describes the PCI bus structure and DPR memory used by the PPC-P11.1
Chapter 21	**BTC06 Interface** Describes how the BTC06 is used with VisualMotion

Table 12-1: Volume 2 Contents

12.3 How to use this Manual

If you are not familiar with VisualMotion software, begin with volume 1 of the Functional Description manual to learn how the software operates. Once you have an understanding of how to get VisualMotion installed and how to create a program, use this volume to find reference information for the features of VisualMotion and IndraLogic as you build your program.

13 Menu Descriptions

13.1 Overview

Access to user programs and system data is provided via menu selections and toolbar buttons. Main menu selections are available in project and service modes. Some menu items available under each main menu selection are dependent upon the current programming mode. Refer to *Programming Modes* in section 4.1 of volume 1 for details.

Some of the more frequently used menu commands are available as toolbar buttons. VisualMotion Toolkit 11 displays menu items and toolbar icons based on the current programming mode. Refer to *Working with VisualMotion Toolkit Icon Palettes* in section 14.1 for details.

13.2 The File Menu

The file menu displays different selections based on the mode of communication (offline/online/service).

The **File** menu commands are as follow:

- standard Windows functions (New, Open and Close)
- saving VisualMotion programs and projects
- switching between online and offline programming modes
- importing and exporting of project components
- printing VisualMotion programs/elements (variables, etc.)
- accessing sample programs and quick launching of recently open programs or projects

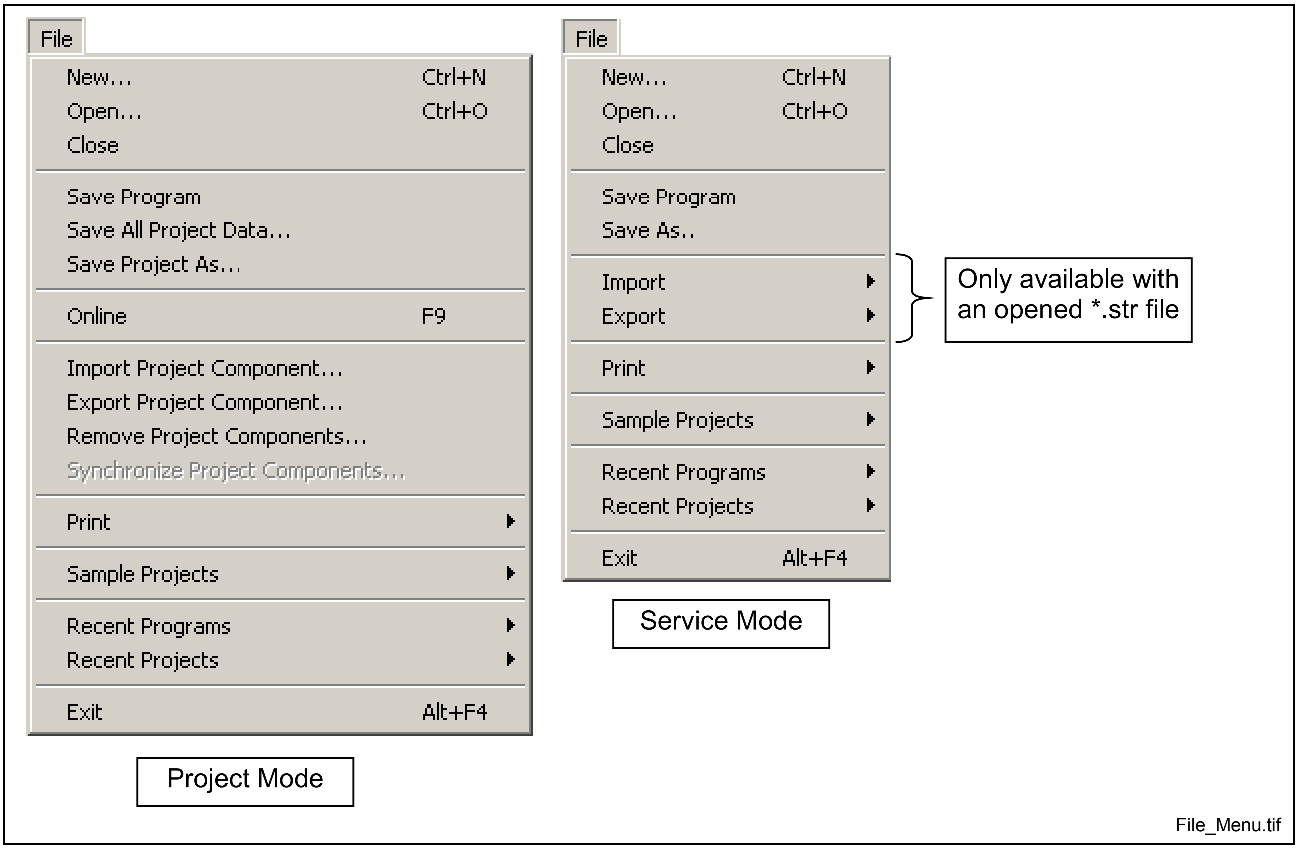


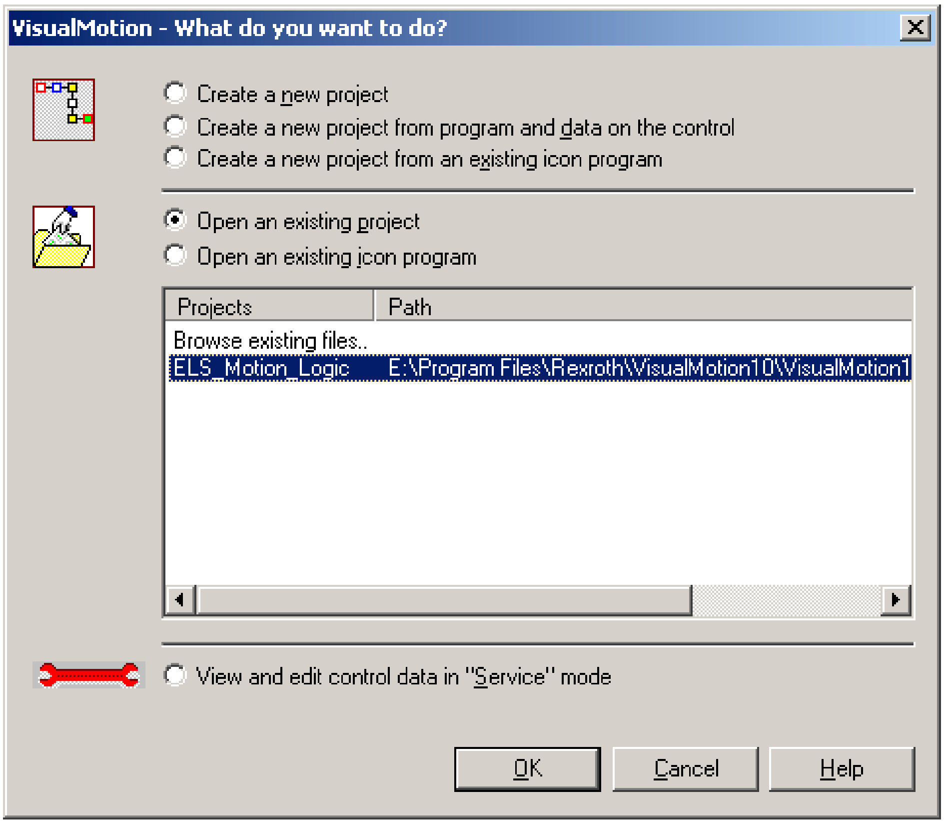
Fig. 13-1: The File Menu

File ⇒ New…

Selecting ***File*** ⇒ ***New*** opens the "*What do you want to do*" window in Fig. 13-2. From this window, the user can select a radio button for one of the following choices.

- Create a new project
- Create a new project from program and data on the control
- Create a new project from an existing *.str program
- Open existing project
- Open existing icon program
- View and edit control data in service mode

Note: If the Cancel button is pressed, VisualMotion Toolkit is launched in service mode.



File_new.tif

Fig. 13-2: New VM Program Development Window

Create a New Project

VisualMotion Toolkit 11 introduces a project structure that organizes all data relevant to an icon program under a specific folder. Any new program, regardless of the target firmware, that is created using VisualMotion Toolkit 11 is created and structured as a project.

When creating a new program, VisualMotion Toolkit 11 creates a project folder, named by the entry in the **Project name** field, where the icon program and all relevant data is stored. The project folder is created under the **Project Location** hard drive designation. The default project location is "\Rexroth\VisualMotion11\project". All modifications made to any component of the project will be maintained in the project folder structure.

Note: The available icons and functionality are based on the selected **Target Firmware**. For example, if GPP8 is selected, the project will not contain an initialization task and only those icons available at the selected firmware level will be available.

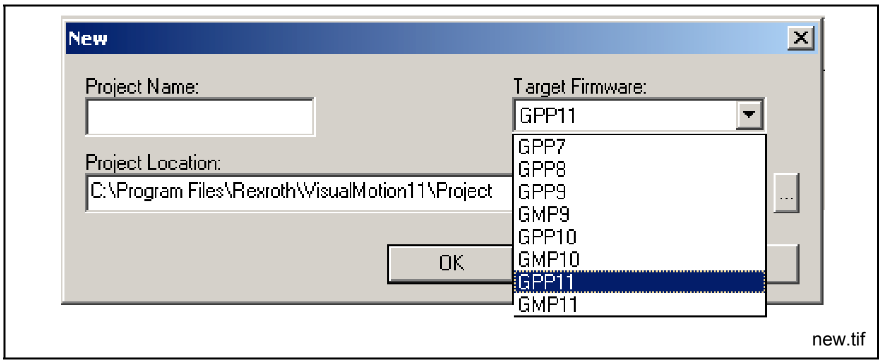


Fig. 13-3: New Project

Project File (vmj)

VisualMotion Toolkit 11 creates a project file and saves it under the "machine1" project folder, i.e., "machine1.**vmj**". The project file contains project specific information about the assigned drives in an icon program used to initialize DriveTop, serial settings, etc. This file allows the user to copy or move the main project folder to any location while maintaining all required settings. Data is saved to the file when the *OK* button is clicked in the "Axis" or "ELS Group" icons, or at compile time. Data saved to the file includes the Sercos address, name and mode of operation of drives. Drives added to the icons and then later removed will remain in the file until the file is rebuilt at compile time.

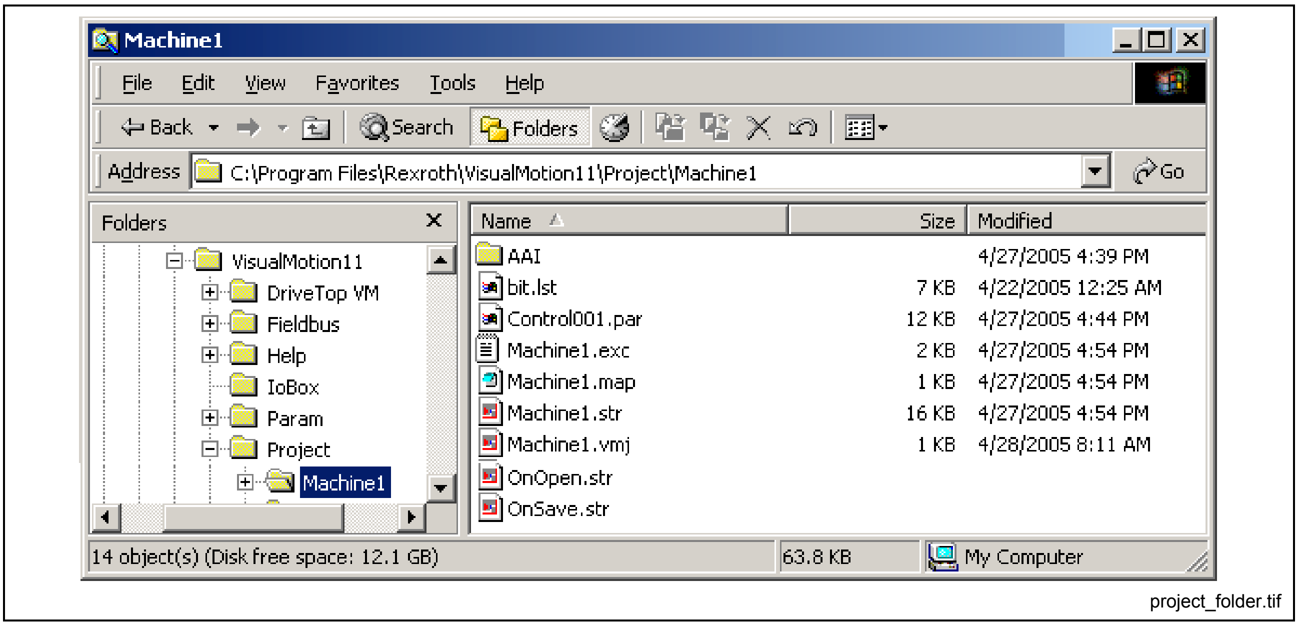


Fig. 13-4: Project Folder

Create a New Project from Program and Data on the Control

A new VisualMotion project is created from the source data retrieved from the control in online mode. When selected, the user selects the communication type (serial or EtherNet) and enters a new project folder name where the source data of the currently active program is stored. The source data consists of the following:

Note: This command can only be performed if an icon program was previously archived to the control's memory from the *Synchronize Project Data* window. Refer to Online / Offline … F9 on page 13-6.

- Icon (*.str) program
- Project (*.vmj) file
- System variables (Floats, Integers, Global Floats and Global Integers)
- Event data
- PLS data
- Points data
- Zone data
- System parameters

Create a New Project from an Existing Icon Program

This menu command is used to open an existing program (*.str) file for creating a new project. A new sub-folder is created containing a new project (*.vmj) and source(*.str) file.

Open Existing Project

This menu command is used to open a VisualMotion project (*.vmj) from a listing of up to 8 previously opened projects or browse to locate a project file on the hard drive.

The project's name and entire path is displayed. The number of displayed project files can be specified, up to a maximum of 8, from the *VisualMotion Options* window by selecting ***Tools ⇒ Options…***.

Open Existing Icon Program

This menu command is used to open a VisualMotion icon program (*.str) from a listing of up to 8 previously opened programs or browse to locate an icon file on the hard drive. VisualMotion 11 will open in service mode.

The program's name and entire path is displayed. The number of displayed program files can be specified, up to a maximum of 8, from the *VisualMotion Options* window by selecting ***Tools ⇒ Options…***.

Note: When opening pre G*P 11 firmware source (*.str) files, the icon and functionality used with the older program is supported in VisualMotion 11.

View and Edit Control Data in Service Mode

This menu command is used to edit data in service mode. The intent of this mode is to provide access to a machine in the field when no project data is available. From service mode, the user can perform backup and restore operations as performed with previous versions of VisualMotion Toolkit.

Open… 

This menu command is used to open any of the following project or program files:

- Project file (*.vmj)
- Icon Motion File (*.str)
- Textual Motion Files (*.mtn)
- Embedded Icon Files (*.exb, *.exc)

Note: Embedded icon files require that they were compiled using the *Include compressed source* option available in VisualMotion 8.

Close

This menu command closes an existing project or icon program and switches VisualMotion Toolkit to service mode.

Save Program

An icon program (*.str), belonging to a project, can be modified and saved back to the project folder while in offline or service mode. When the project file (*.vmj) is opened and VisualMotion Toolkit is switched to online mode, all modifications are automatically detected and the user is ask to Save, Compile and Download changes.

Note: This menu command is only available in offline or service modes.

Save As…

This menu command is used to save programs and data when working in service mode.

Note: This command is only visible in the File menu when working in service mode.

Save All Project Data…

This menu command saves the current project data to the appropriate project folder location when modifications are made in offline mode. Any tools and utilities that have been modified and are currently opened will also be saved.

Save Project As…

This menu command copies project files from the source project folder to location specified in the **Project Location** field. The target firmware type can also be specified before saving the project. Only files relevant to the project are copied.

Note: This command is only available from a project in offline mode.

Online / Offline … F9

A project is switched to online or offline mode by clicking the **Toggle Online/Offline** icon toolbar button, selecting this menu selection, or by pressing the **F9** key. This initiates the VisualMotion synchronization process. Refer to *Synchronizing a Project* in section 10.1 for details.

Import Project Component…

Project components can be imported from another project or from a file when in offline mode. In online mode, project components are imported from the control's memory.

For specific information on how to import project components, refer to section 10.4 in volume 1.

Note: In service mode, an **Import** menu selection is available when a valid *.str file is opened and allows the user to import a valid Variable, Register, or Bit label file into the current project.

Export Project Component

Project components can be exported to another project or to a file when in offline mode. In online mode, project components are exported to the control's memory.

For specific information on how to export project components, refer to section 10.5 in volume 1.

Note: In service mode, an **Export** menu selection is available when a valid *.str file is opened and allows the user to export a Variable, Register, or Bit label file from the current project.

Remove Project Components

Project components can be removed from the project currently opened while in offline mode. Components can not be removed in online mode.

For specific information on how to remove project components, refer to section 10.6 in volume 1.

Synchronize Project Components

The synchronization of project components is available in online mode when modifications to the current project are made and saved.

Refer to *Synchronizing a Project* in section 10.1 for details.

Print Project Data…

Selecting ***File*** ⇒ ***Print ...*** from the File menu displays the Documentation Selection window. All **Program Data** is selected by default.

Program Data

Selecting **Icon Data Flow** provides a graphic printout of a VisualMotion Icon Language window. For this function, you must have a graphics-capable printer. Text-based files, such as Text Language user programs or parameter files uploaded for viewing, may also be loaded and printed from Windows Notepad or another editing program.

Setup Information can be printed for "All" or a selected task, subroutine, or event.

Selecting **Variable Labels**, **Register Labels**, **Bit Labels, Other References and/or Subroutine List** prints a list of the respective labels for the task, subroutine, or event window currently displayed by VisualMotion.

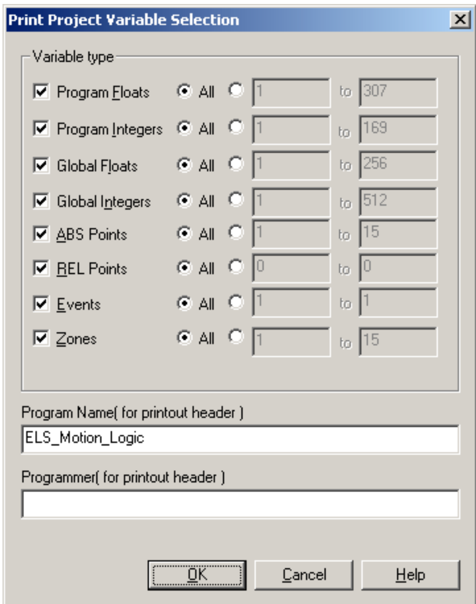
The fields for **Project Name** and **Programmer** allow for adding the project name and programmer to the header on each printed page. The standard header contains the filename, task name, date, time and page number. Date and time are relative to the time of printout and are based on the time kept by the PC.

Program

Selecting ***File*** ⇒ ***Print*** ⇒ ***Program…*** opens the *Documentation Selection* window where the user can select which program data to the print.

File Data

Selecting ***File*** ⇒ ***Print*** ⇒ ***File Data…*** opens the *Print Project Variable Selection* window. The user has the option to select All the variables of a specific type or specify a range to print. The printout displays the ID, Label and value for all variable types selected.



print_var_data.tif

Fig. 13-5: Print Variable Data

Sample Projects

As part of VisualMotion Toolkit, four sample projects ("Press Feed", "Coordinated Mode", "Flow Wrapper", and “Position Mode" have been included.

Recent Programs

This menu selection displays up to the last 8 (*.str) icon programs that the user can choose from to quickly launch the program in service mode. The number of recent program files can be specified, up to a maximum of 8, from the *VisualMotion Options* window by selecting ***Tools ⇒ Options…***.

Recent Projects

This menu selection displays up to the last 8 (*.vmj) project programs that the user can choose from to quickly launch the project. The number of recent project files can be specified, up to a maximum of 8, from the *VisualMotion Options* window by selecting ***Tools ⇒ Options…***.

Exit

Exits the current program, prompting the user if it has not been saved.

13.3 The Edit Menu

The **Edit** menu contains Windows editing features for icon-based programs, clearing the icon workspace, find/replace functions and view and adding of VisualMotion data types.

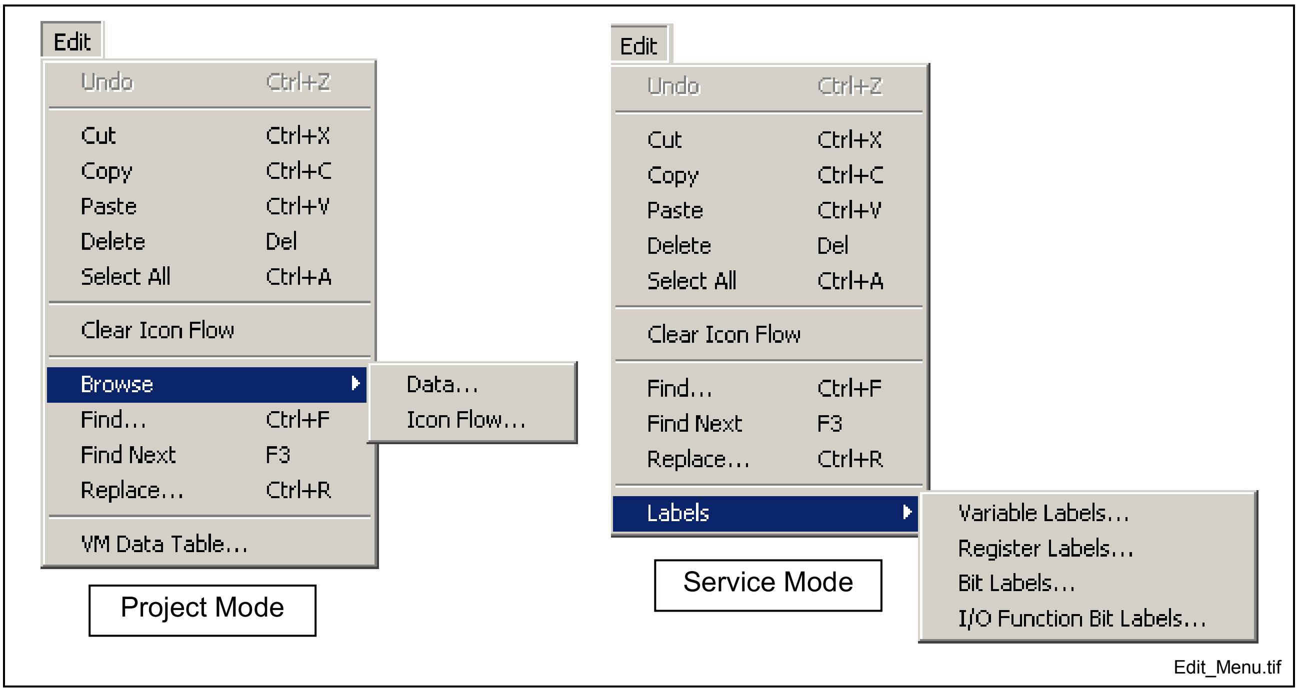


Fig. 13-6: The Edit Menu

Windows Editing Features

The Windows editing features are used to manipulate icons with the user program.

Undo (Ctrl+Z) 

This menu command can undo the last icon edit.

Cut (Ctrl+X) 

This menu command cuts the selected program flow to the paste buffer.

Copy (Ctrl+C) 

This menu command copies the selected program flow to the paste buffer.

Paste (Ctrl+V) 

This menu command enables the paste operation into the selected program space.

Delete (Del)

This menu command deletes the selected program flow from the program space.

Select All

This menu command selects the entire program flow of the current task, subroutine or event function.

Clear Icon Flow

This menu selection deletes all contents of the current VisualMotion task, subroutine, or event workspace of the open program. The Undo menu selection or icon can be used to undo only the last modification.

Browse

The browse feature in VisualMotion 11 allows the user to locate usage of program data in a project and usage of subroutines and event functions in the icon flow.

For specific information on how to use Browse, refer to section 10.2 in volume 1.

Find, Find Next, Replace

These menu items are used to locate variables or subroutine/event functions in the open program.

Find…

Selecting ***Edit*** ⇒ ***Find…*** searches for the first occurrence of specified text and opens the following window:

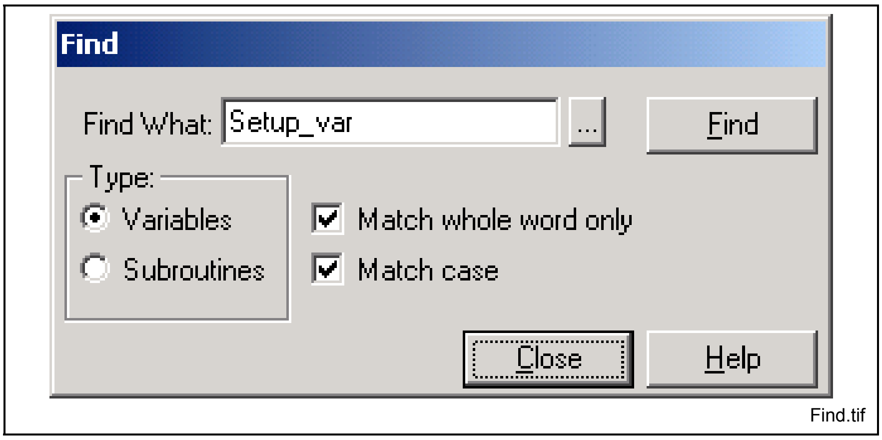


Fig. 13-7: “Find” Window

1. Select **Variables** or **Subroutines** in the **Type:** grouping.
2. Enter the text in the **Find What:** field. The browse button provides a selection list of all variables or subroutines found in the open program.
3. Check **Match whole word only** or **Match case** to limit the search.
4. Click **Find** to locate the first occurrence of the text.

Note: Only variables and subroutines that are used in the program are found. The VM Data Table may contain variables that are not used in the program.

Find Next

Selecting ***Edit*** ⇒ ***Find Next*** (or the **F3** key) searches for the next occurrence of the same text designated for the last ***Find*** operation.

Replace…

Selecting ***Edit*** ⇒ ***Replace*** allows for searching and replacing specified text and opens the following window:

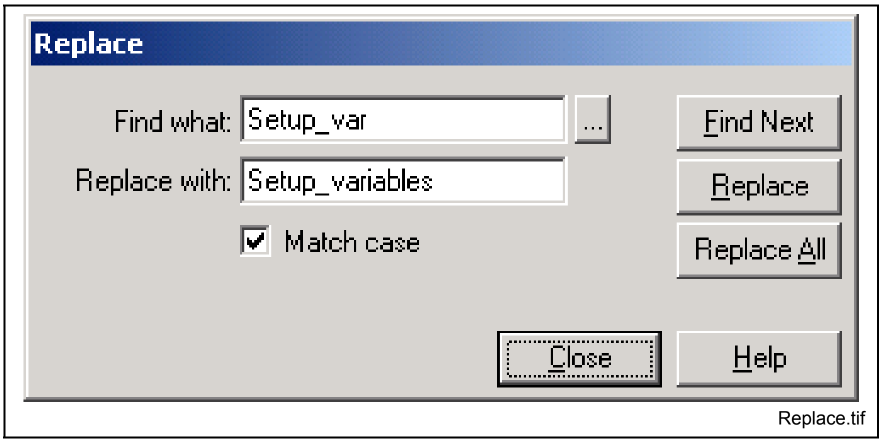
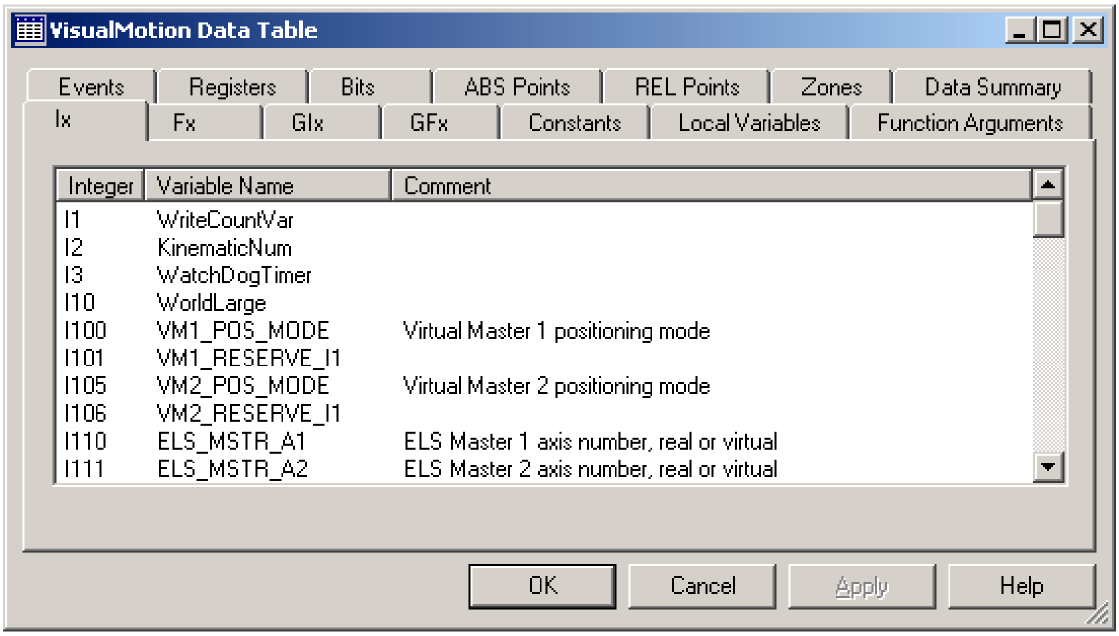


Fig. 13-8: “Replace” Window

VM Data

The VM Data Table is used to Add, Delete, Edit, Find, Browse, Shift, Import and Export the following VisualMotion data types.



VMdataTable.tif

Fig. 13-9: VisualMotion Data Table

For specific information on how to use the VM Data Table, refer to section 4.12 in volume 1.

Labels

The Labels menu selection is only available in service mode when a valid program file (*.str) is opened in VisualMotion Toolkit. It's used to add, delete, or edit labels in the currently opened program.

For specific information on how to use the Labels menu selection, refer to section 4.13 in volume 1.

13.4 The View Menu

The **View** menu allows the user to select a task (Initialization or A-D), subroutines, event functions for display or editing, and allows viewing nested subroutines. The user can also select pre-configured icon palettes for single axis, coordinated, Electronic Line Shafting (ELS) and Utility.

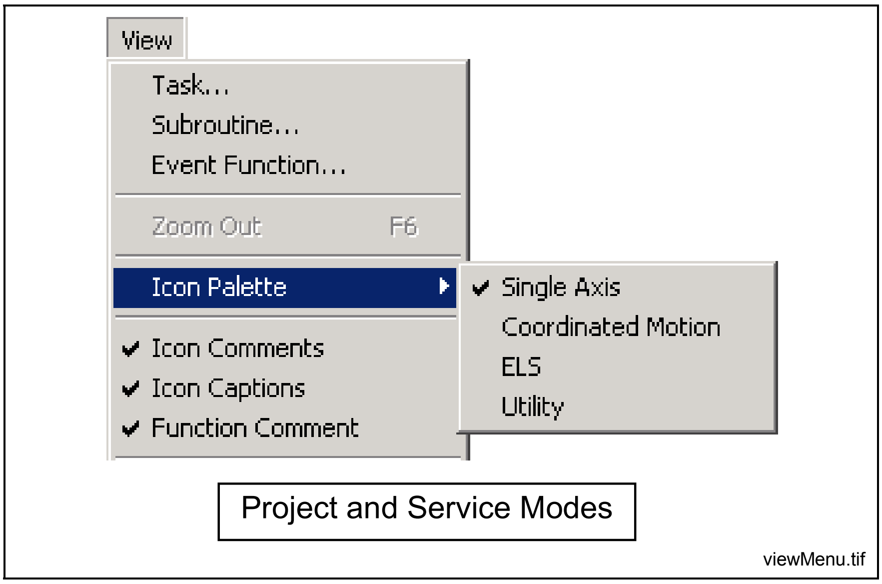


Fig. 13-10: The View Menu

Task…

Selecting ***View*** ⇒ ***Task*** displays the System Tasks Window. This window is used to navigate between the different task available in the system.

Note: VisualMotion task can also be viewed by using the Project Navigator.

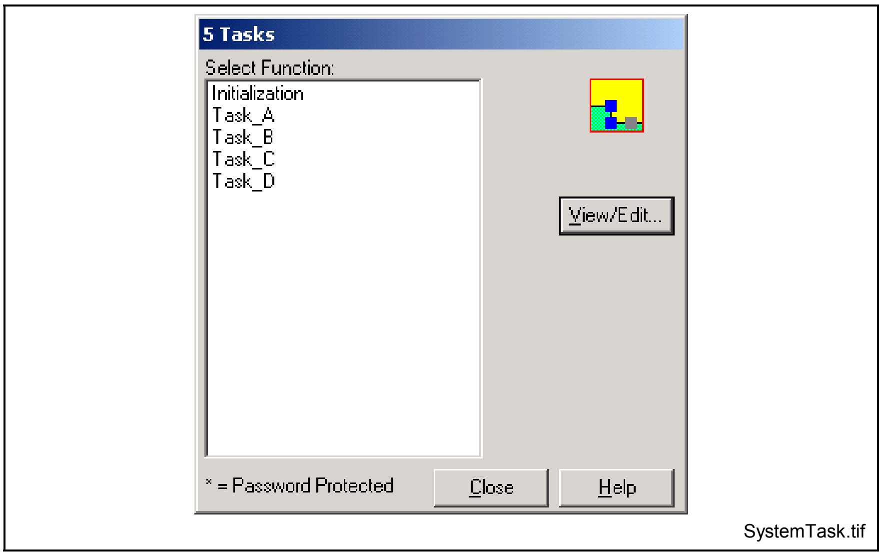


Fig. 13-11: System Tasks

The initialization task is used to configure all icons that are initialized during the Sercos Phase 2 to Phase 3 transition. The icon palette to the left of the workspace is displayed with only those icons that are initialized in Phase 2. Icons such as the Axis setup or ELS setup icons are place between the start and finish icons.

Note: The single axis, coordinated motion, ELS and Utility icon palettes are **not** accessible from the Initialization task.

When Task A – D are selected, icon palettes not accessible in the Initialization task can be selected from the View menu or by clicking the different icon tabs at the bottom of the displayed icon palette.

Note: The initialization icon palette is not visible when in task (A-D).

Subroutines

Selecting ***View*** ⇒ ***Subroutines*** displays the subroutine window. Only subroutines that were created using ***Insert*** ⇒ ***Add Subroutine*** or the subroutine icon () can be viewed. Subroutines can be viewed for both the Initialization and Task A-D. Password protected subroutines are displayed with an asterisk.

Note: VisualMotion subroutines can also be viewed by using the Project Navigator.

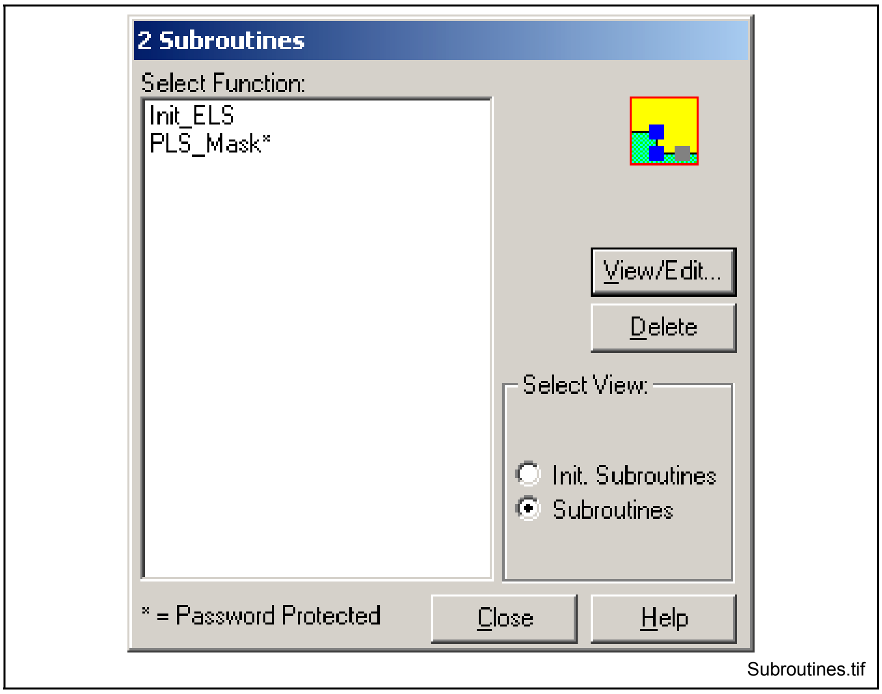
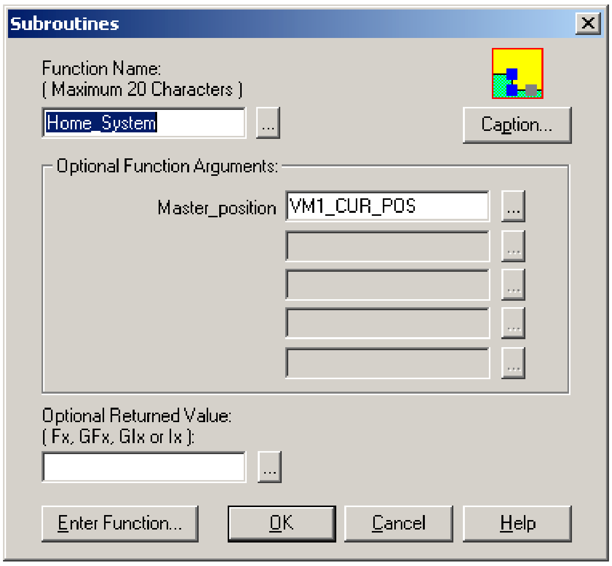


Fig. 13-12: “Subroutines” Window

The programmed subroutines are listed by name with the total number listed in the title bar. Selecting a subroutine from the list permits deleting or replacing the current VisualMotion workspace with the selection.

A subroutine can also be viewed in the workspace by double clicking an existing subroutine icon to display the **Subroutines** icon window. Refer to the **Sub1** icon description in chapter 14 for details.



Subroutine_Icon.tif

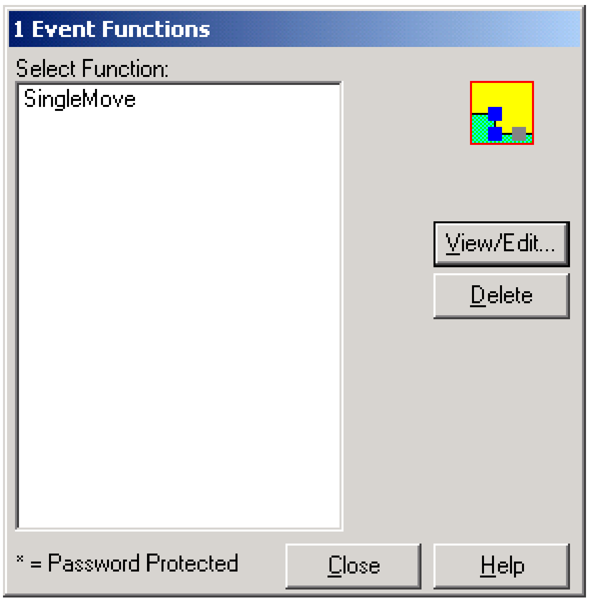
Fig. 13-13: “Subroutines” Icon Window

In this window, optional arguments and an optional returned value could be entered. The subroutine's icon name is displayed as the default. Clicking **OK** exits the window. Clicking **Caption** displays another window, which permits entering/editing of the icon caption and comment text. Clicking **Enter Function…** displays the subroutine in the VisualMotion workspace.

Event Functions

Selecting ***View*** ⇒ ***Event Functions*** displays the Event Functions window. Only events that were created with ***Insert*** ⇒ ***Event Function,*** the event icon ( ) or the VM Data Table icon ( ) can be viewed. Password protected subroutines are displayed with an asterisk.

Note: VisualMotion event functions can also be viewed by using the Project Navigator.



event.tif

Fig. 13-14: “Event Function” Window

The programmed event functions are listed by name in a selection list and the total number is listed in the title bar. Selecting an event function from the list permits deleting or replacing the current VisualMotion workspace with the selection.

Note: Refer to section 5.2 in volume 1 for examples of how to use the Event icon.

Zoom Out F6

Selecting ***View*** ⇒ ***Zoom Out F6*** is used to load the VisualMotion workspace with the parent (the task or other subroutine from which the current subroutine is called) of the presently loaded subroutine.

Note: The ***Zoom Out*** command is only available when entering a function using the ***Enter Function*** button within the subroutine icon. In all other cases, the function appears grayed out in the menu selection.

Zoom Out, (or the **<F6>** key), is used to move back from a nested subroutine queue, one subroutine at a time. The queue of nested subroutines is displayed in the title bar of the VisualMotion workspace.

Example:

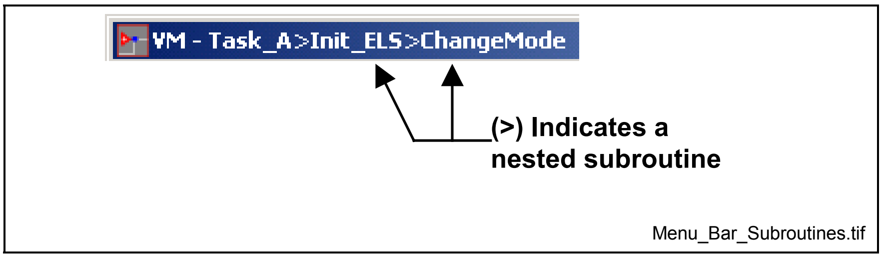


Fig. 13-15: Nested Subroutines in VisualMotion Menu Bar

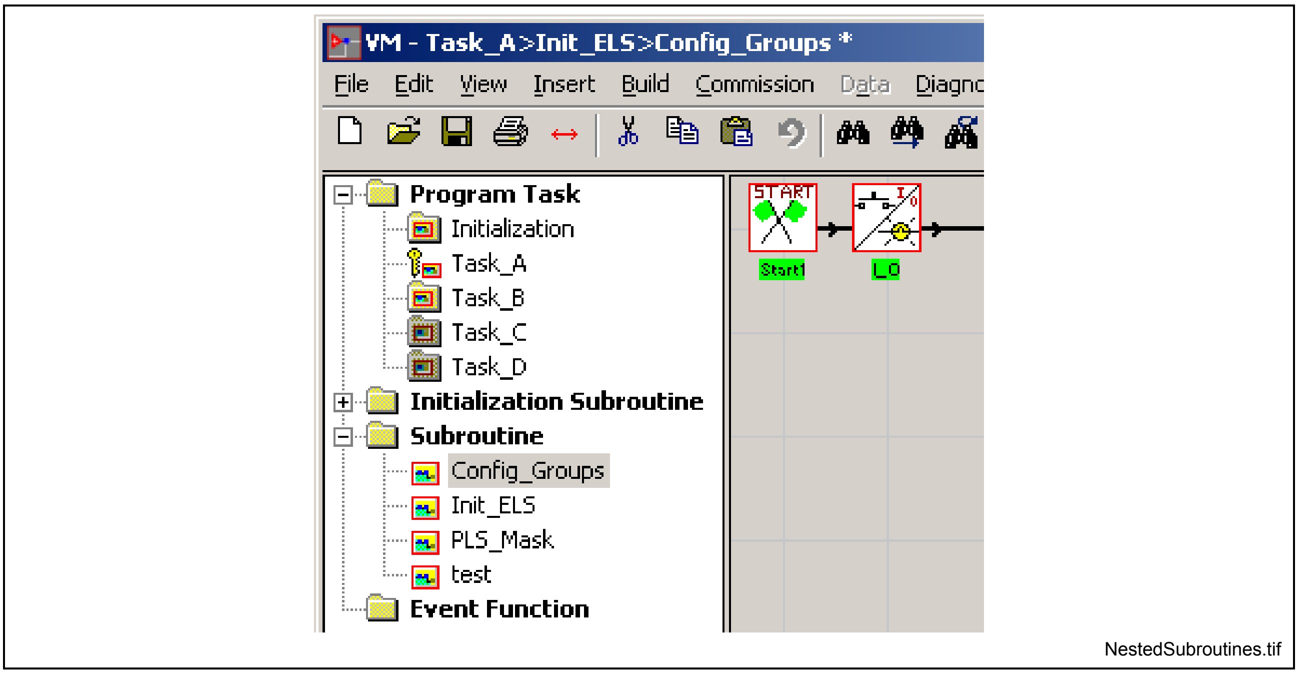


Fig. 13-16: Nested Subroutine Queue in the VisualMotion Menu Bar

The ***Zoom Out*** command is unavailable when any of the four main tasks (A - D) or Initialization task are loaded in the workspace. When the presently loaded subroutine is opened via the ***Subroutines*** menu command instead of the **Enter Function…** button. Since event functions are independent and invoked by a specified axis motion or time-based program function, they have no "parent". Therefore, ***Zoom Out*** cannot be used to return to a higher level from an event.

Icon Palette

VisualMotion Toolkit 11 icon palettes are displayed below the Project Navigator. Each icon palette can be selected by clicking on the small icons at the bottom of the icon palette or by selecting ***View*** ⇒ ***Icon Palette***.

Note: The icon palette menu selection is only available in offline and service mode when any Task (A-D), subroutine or event function is selected in the Project Navigator.

Refer to Icon Descriptions in chapter 14 for details.

Icon Comments

Selecting ***View*** ⇒ ***Icon Comments*** enables/disables identifying comments that appear when the mouse cursor is moved over the top of the icon. The icon comments in a program flow can be modified in the corresponding icon setup window. The setup window opens when the icon is first placed or by double clicking, the left mouse button, while the pointer is over the icon. Each window has a "Label" button to edit its comment text, by default there is no comment. The comment also appears on the printout of the setup information, if enabled or not.

Icon Captions

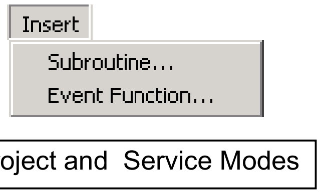
Selecting ***View*** ⇒ ***Icon Captions*** turns the icon labels in the VisualMotion workspace on or off. If there are no user entered labels, VisualMotion uses the default icon labels.

Function Comment

Selecting ***View*** ⇒ ***Function Comment*** enables and disables the comment header for the current task. Each VisualMotion task contains a comment header section below the icon programming workspace where the user can describe any aspect of the task. Any standard ASCII character can be used, up to a maximum of 32K.

13.5 Insert Menu

The Insert menu is available in project and service modes and is used to add subroutines and event functions to a VisualMotion project.

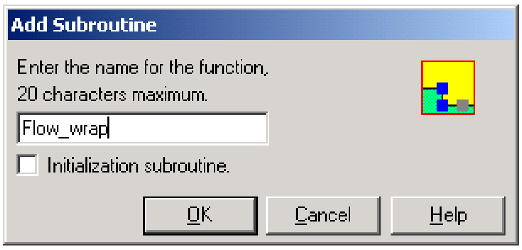


InsertMenu.tif

Fig. 13-17: Insert Menu

Subroutine

Selecting ***Insert*** ⇒ ***Subroutine*** opens the *Add Subroutine* window. This menu item is used to add a subroutine within a task. Subroutines are called out within the program flow and used to program functions or routines that are used within a task or program. For example, a subroutine can be used to calculate certain values before the program proceeds. Any calculation can be programmed in a subroutine, making the main task icon program less congested.



Add_Subroutine.tif

Fig. 13-18: “Add Subroutine” Window

Adding a Subroutine

The subroutine menu selection is used to open a new window where the actual subroutine function (icon program) is created. Once created, the desired program flow location from where the subroutine function will run is chosen by placing a ***Subroutine Icon*** and selecting the name of the created subroutine function.

Selecting a Name

To create a subroutine function, enter a name for the subroutine. This name will appear when a subroutine icon is placed in the program flow of the desired task. The name has a 20-character limitation and must begin with an alpha character with no spaces. Pressing the **OK** button opens a subroutine workspace with the name entered, a **Start** and **Finish** icon in place. You may then write a subroutine function using icons and connecting lines in the same manner as when writing an icon program task.

Initialization Subroutine

The initialization subroutine checkbox is used to identify whether or not the subroutine function will be used in the initialization task or in one the 4 standard tasks (if left unchecked). The icons that are allowed for placement in the subroutine function depend on the initialization subroutine checkbox. If checked, only Phase 2 (initialization) icons are allowed. If left unchecked, then the icons available in the Single Axis, Coordinated, ELS and Utility palettes are available for placement.

Note: A maximum of 500 screens, consisting of tasks, subroutines and event functions are possible.

Event Function

Selecting ***Insert*** ⇒ ***Event Function*** opens an *Event Function Control Block* window.

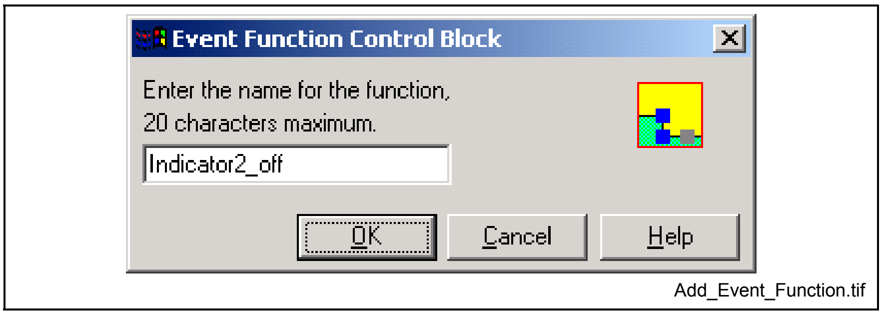


Fig. 13-19: “Event Function Control Block” Window

Unlike subroutines, event functions are not "called" from a program. Instead, they are "triggered" by the conditions (distance, time, etc.) that are specified in an event setup. Refer to the ***Calc Icon*** and ***Event Icon*** in Icon Programming chapter for details.

Note: An event function must be written before it can be assigned to an event trigger.

Entering a name for an event function and clicking **OK** opens an event function workspace with the name entered and the **Start** and **Finish** icons in place. You may then write an event function using icons and connecting lines in the same manner as when writing an icon program task.

Note: A maximum of 500 screens, consisting of tasks, subroutines and event functions are possible.

13.6 The Build Menu

The **Build** menu contains commands for compiling and managing VisualMotion user programs.

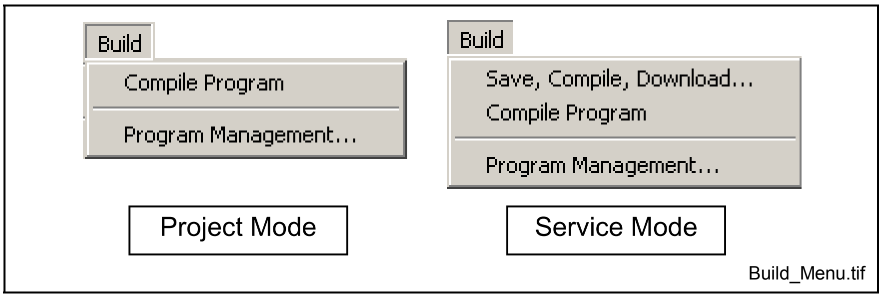


Fig. 13-20: The Build Menu

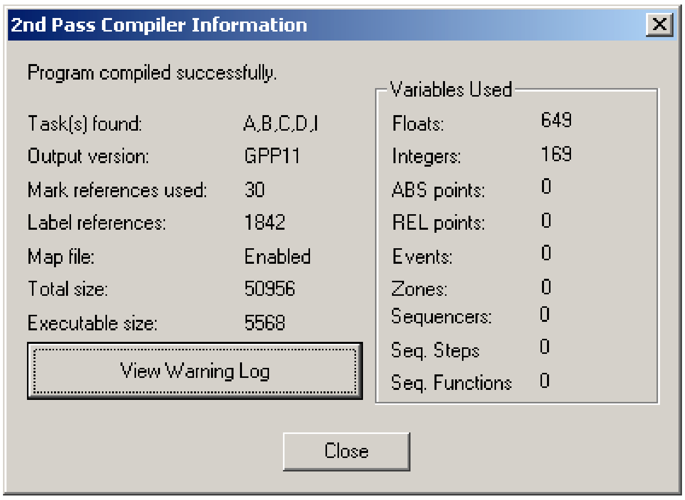
Save, Compile, Download

Saves the currently open program, compiles it, and downloads it to the control. The Program Management window opens when all three functions have successfully been carried out.

Note: This menu item is only available when VisualMotion Toolkit is in service mode.

Compile Program

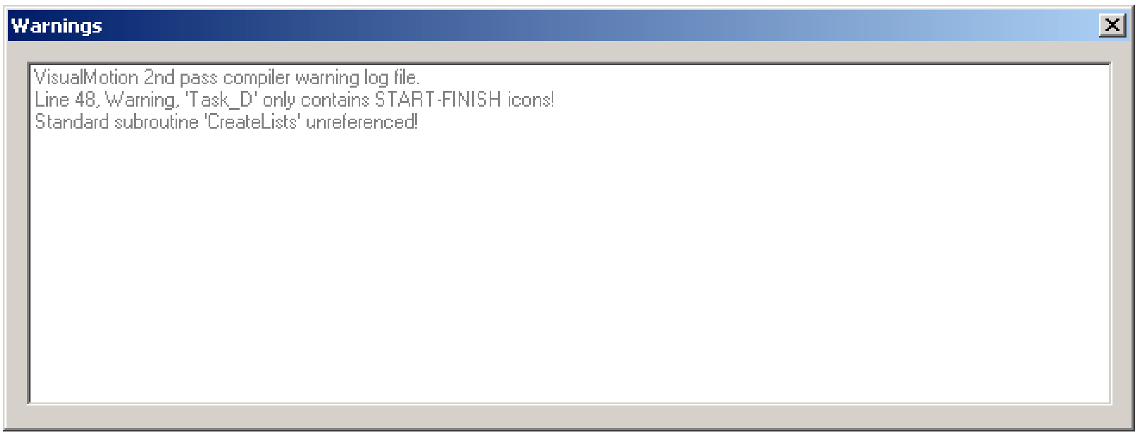
This selection checks and converts the current project to code that the control can interpret. When complete, the 2nd Pass Compiler opens the following window relating information about the compiled program. All component files of the project are saved to the project folder.



CompilerInfo.tif

Fig. 13-21: Compiler Information

The **View Warning Log** button opens the 2nd pass compiler *Warning* window listing any warning that were encountered as well as any subroutines that exists in the project but are not currently being referenced.



CompilerWarning.tif

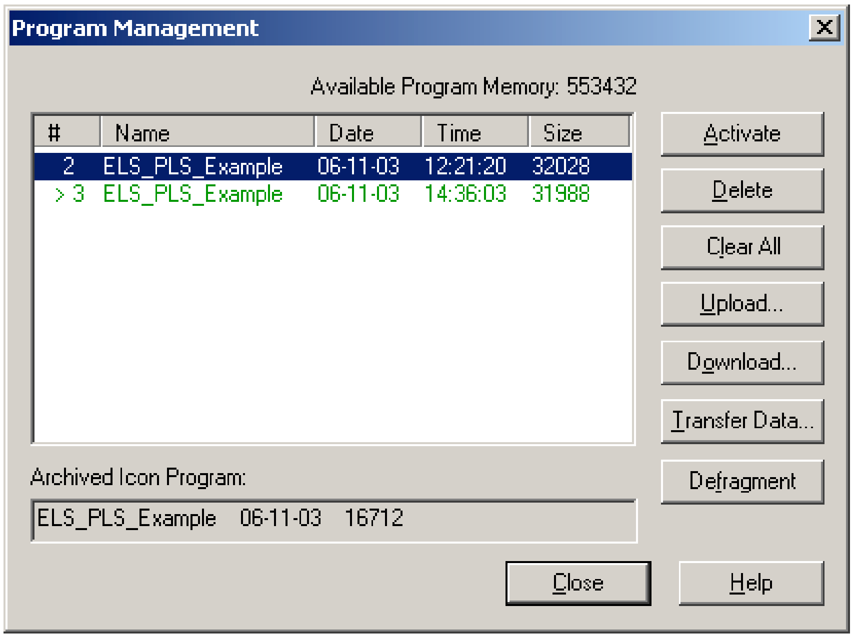
Fig. 13-22: 2nd Pass Compiler Warning

Rexroth Bosch Group

Program Management

The Program Management window is used to activate and download VisualMotion user programs to the control or to transfer data between programs in the control's memory. A maximum of ten (10) user programs can be downloaded to the control with only one activated at any give time. The currently active program name is displayed in green text with a (>) symbol to the left of the name.

Note: This menu selection is available only while in online project mode or service mode.



Program_Management.tif

Fig. 13-23: “Program Management” Window

Note: The PPC-R control uses a flash file system for storing user programs. When a new program is activated, the old program is removed from local memory and stored to flash. Every activation of user program reduces the amount of flash memory. This process causes flash memory fragments. When the flash memory fragment threshold is reached, VisualMotion automatically runs a defragmentation process to restore unused memory.

The Program Management window provides the following buttons:

Activate

This button is used to active a different program previously downloaded to the control’s memory. By default, the currently active program is selected and highlighted. A different program cannot be activate when the active program is running.

Note: Communication errors can occur when a program is activated that performs parameter transfers (Param1) or parameter initializations (Setup Parameter) to X10 or X16 serial port communication parameters. Refer to *Communication Errors After Program Activation* in section 10.3 under heading *Online Full Restore* for details.

Delete

This button deletes the highlighted program from the control's memory. A confirmation window opens before a program is deleted. Active programs can not be deleted from the control's memory.

Clear All

This button erases all resident programs and data from the control's memory (including the active program) when the system is in parameter mode. A confirmation window opens before all programs are deleted. Refer to section 10.3 in volume 1 for archiving project data before clearing programs from the control's memory.

Upload…

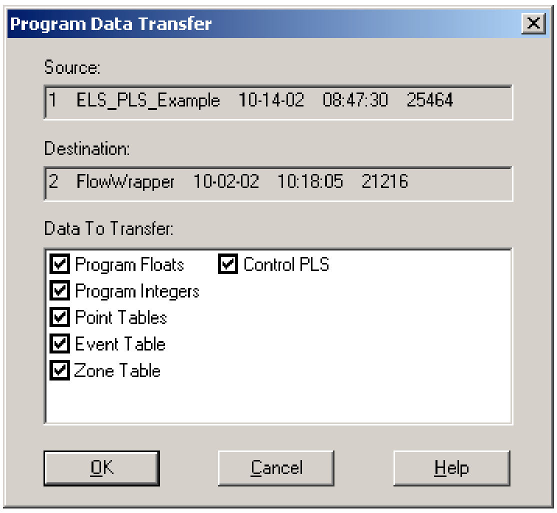
This button uploads the highlighted program for archiving on the host system's hard drive. When prompted with the **Save As** window, select the file destination and filename. A progress bar indicates the upload status.

Download…

This button downloads a compiled program to the control's memory. After clicking the **Download** button, select or type the desired filename, and then click the **Open** button. Executable programs are stored together with other project files in the project Directory. After clicking **OK**, the download is executed, and a progress bar shows the download status.

Data Transfer

This button transfers data items between the active program (Source) and the (Destination) program selected for activation. This button becomes active when a program, other than the active program, is selected. Clicking on the **Data Transfer** button opens the Data Transfer window.



Data_Transfer.tif

Fig. 13-24: Selective Data Transfer

Clicking on **OK** copies all the data items from the control's active program to the target program selected from the Program Management resident program list.

Clicking on **Selective Transfer** allows the user to selectively choose the following data types:

- Floats
- Integers
- Absolute and Relative Points
- Events
- Zones
- Sequencers
- Control PLS

CAUTION

Data Transfer overwrites the data sets of the destination program.

⇒ If the source program data allocation (i.e., the amount of configured data types) is larger than the target, then only the data elements within the "VM Data" allocated by the target are transferred.

For Example:

If the active program has 75 events and the selected target program has a only 50 events specified in the VM Data Table, only the first 50 events of the active program will be transferred. Data Transfer is a useful tool for developing programs by incrementally testing and modifying sequential copies of a working program without the need to continuously re-input data sets for the new program.

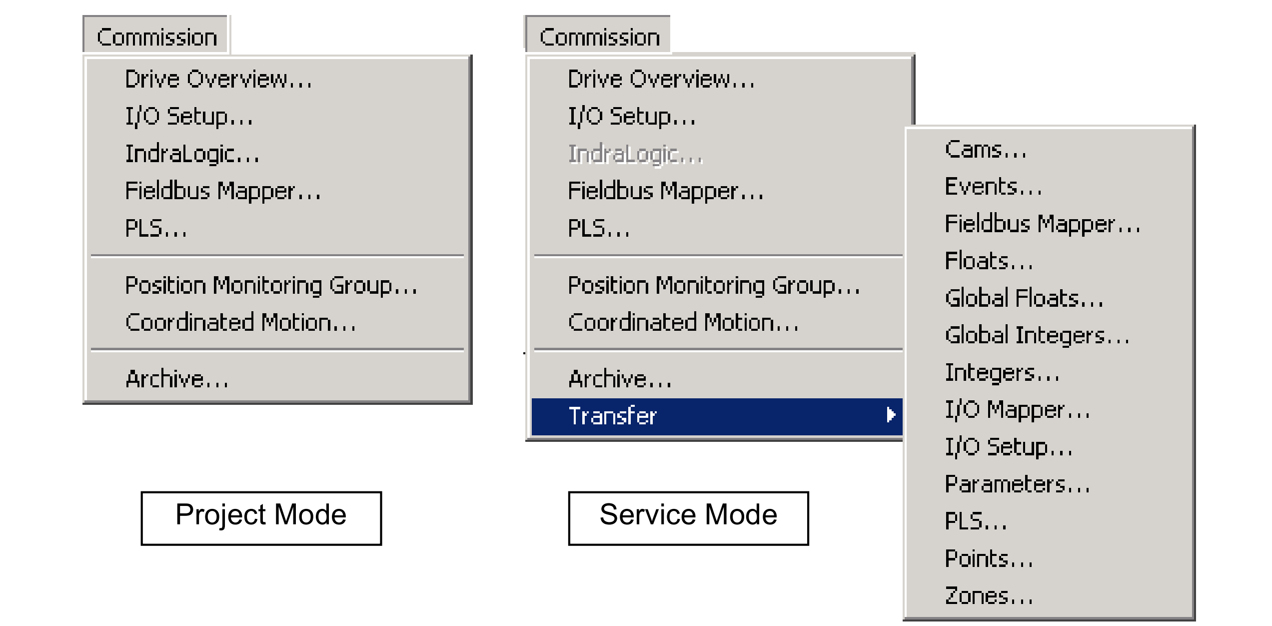
Defragment

This button is used to force a flash compression of unusable control memory. Unusable memory is created when programs and control data is updated or cleared. This low priority task runs in the background. The status bar indicates the progress of the defragmentation.

13.7 The Commission Menu

The **Commission** menu contains tools for setting up and configuring program related interfaces, such as drives, I/O, fieldbus interfaces and PLS functionality. The user can also configure Position Monitoring Groups and Coordinated Motion requirements, archive and transfer program data.

Note: The Transfer menu selection only appears when VisualMotion Toolkit is in service mode.



Commission_Menu.tif

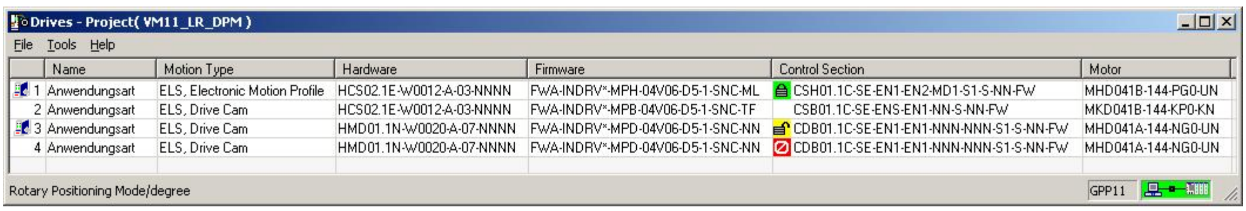
Fig. 13-25: The Commission Menu

Drive Overview

Selecting ***Commission*** ⇒ ***Drive Overview*** opens the drive overview window. This window displays all the digital drives detected on the Sercos ring and displays basic overview information on each drive.

The *Drive Overview* window can also provide the following information:

- Detection of installed and activated IndraDrive Safety Technology
- Detection of drives configured using DriveTop
- Hardware and/or firmware differences between drives detect on the Sercos ring with those setup in current project



DriveOverviewSafety.tif

Fig. 13-26: Drive Overview

Double clicking on a listed drive opens the *DriveTop Drive Status* window.

Rexroth
Bosch Group

IndraDrive Safety Technology Support

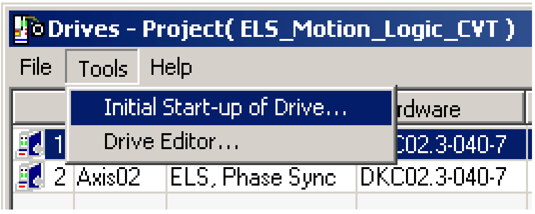
The *Drive Overview* window supports the detection of IndraDrive safety technology in online mode. When detected, the *Control Section* column displays an icon next to the control section description showing the current state of the installed safety components. Refer to the following table for details.

Control Section Icon	Description
Green and Locked	Safety activated and locked (P-0-3207 = 2)
Yellow and Unlocked	Safety activated but not locked (P-0-3207 = 1)
Red and Locked	Safety installed, not active and not locked (P-0-3207 = 0)
No icon displayed	No safety installed

Table 13-1: IndraDrive Safety Technology Detection

Drives Configured Using DriveTop

Any drive that is initialized to its base parameter set using the *Initial Start-up of Drives…* option will display a small DriveTop icon next to its assigned address number.



InitialStartUp.tif

Fig. 13-27: Drive Initialized to Base Parameters using DriveTop

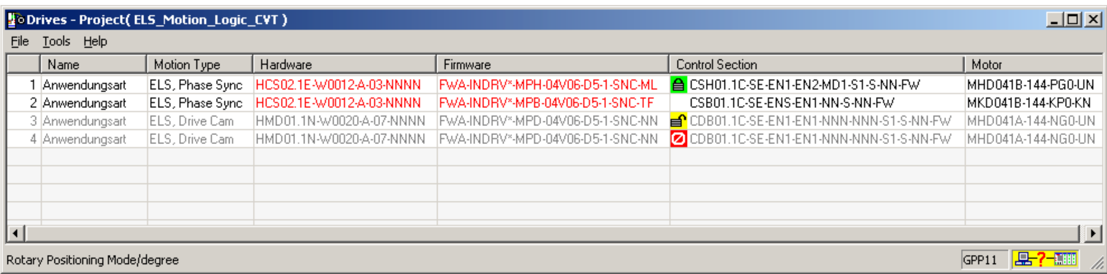
The *Initial Start-up of Drives* option can be selected in online mode from the *Tools* menu in the *Drive Overview* window, by right clicking over an axis under *Setup* ⇒ *Axes*, or under the *Setup* menu in DriveTop.

Differences Detected

In online mode, the *Drive Overview* can detect drive hardware and/or firmware differences between the current project and the drives found on the Sercos ring. These differences are displayed as follows:

- Red text indicates a difference in drive hardware and/or firmware
- Gray text indicates that the drive found on the Sercos ring does not exist in the current project

In service mode, only drives detected on the Sercos ring are displayed. No comparisons with drives in a project can be performed.

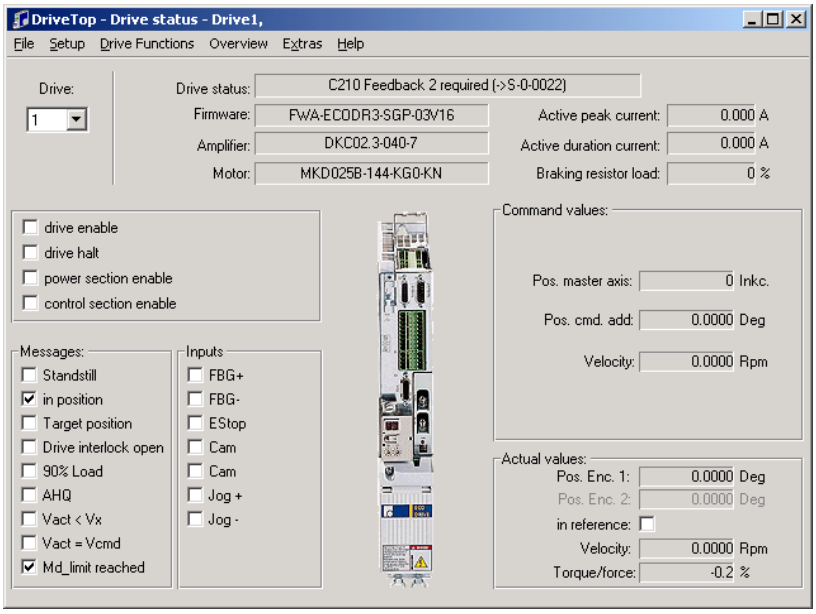


DriveOverviewSafetyDifference.tif

Fig. 13-28: Difference Detection

DriveTop Drive Status

Selecting the *Overview* button will opens the *DriveTop Status* window in Fig. 13-29. All drive setup windows initially configured using the *Commission* button can be selected from the menu items found in the *DriveTop* status window.



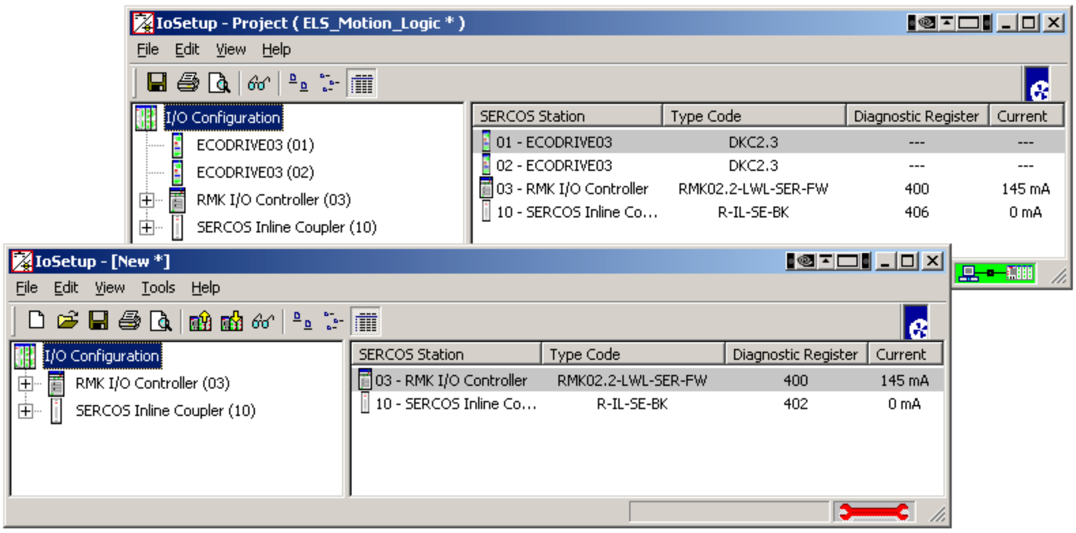
DtopDriveStatus.tif

Fig. 13-29: DriveTop Drive Status

For specific information on how to use DriveTop, refer to section 7.4 in volume 1.

I/O Setup

Selecting ***Commission ⇒ I/O Setup*** opens the I/O Setup Tool window in Fig. 13-30. This tool is used to configure Sercos based I/O devices supported by VisualMotion.



IO_Config_Tool.tif

Fig. 13-30: I/O Setup Tool Window

For specific information on how to commission I/O, refer to section 7.2 in volume 1.

IndraLogic

Selecting ***Commission ⇒ IndraLogic*** opens IndraLogic. IndraLogic is a complete development environment for declaring all Local RECO I/O and PLC programs.

For specific information on how IndraLogic is integrated with VisualMotion, refer to section 7.1 in volume 1.

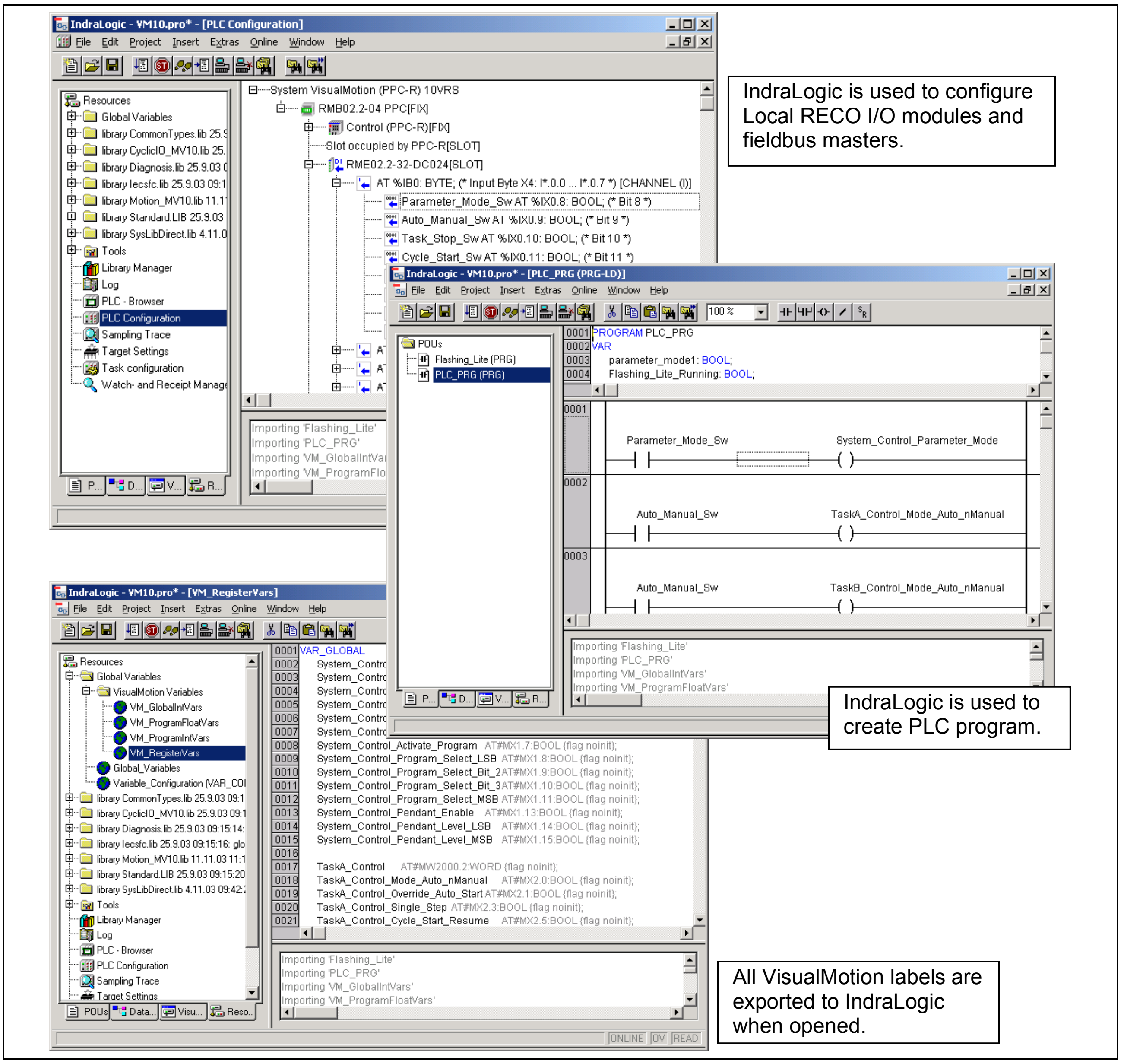


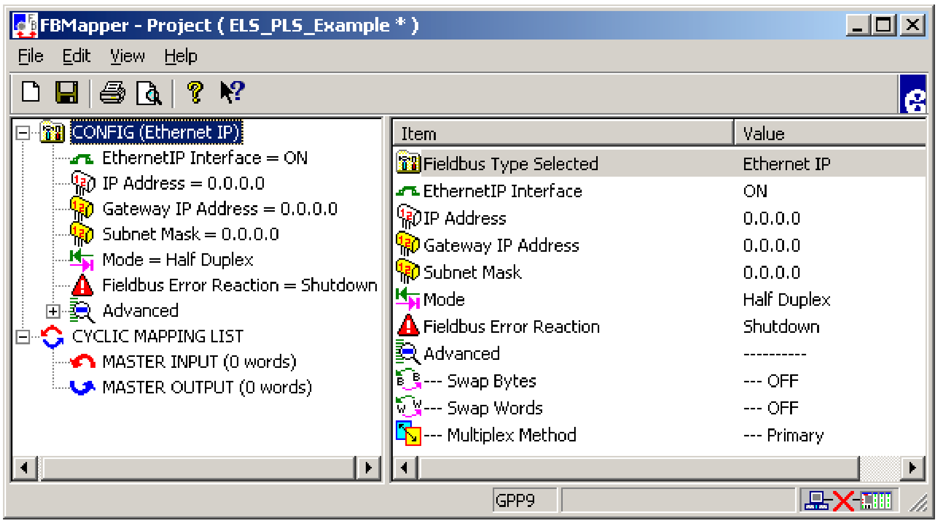
Fig. 13-31: IndraLogic PLC Editor

Note: When opening projects created in pre G*P 11 firmware, the I/O Mapper is launched. Refer to the VisualMotion 9 Functional description for I/O Mapper details.

Fieldbus Mapper

The Fieldbus Mapper is used to setup fieldbus configuration and data mapping. Using this tool, the user can map data to and from the control, and save specific mapping lists as a file or download/upload these lists from/to the control.

The following figure shows the main Fieldbus Mapper window:



Fieldbus_Mapper.tif

Fig. 13-32: Fieldbus Mapper Main Window

The available slave fieldbus types for VisualMotion 11 are:

- Profibus
- DeviceNet
- ControlNet
- EtherNet/IP
- Interbus

The data types that can be mapped include:

- Variables (Integer, Global Integer, Float and Global Float)
- Parameters (Axis, Card and Task)
- Registers

For specific information about each fieldbus and for directions on how to map data, refer to volume 1 *sections: 7.6 Profibus Fieldbus , 7.7 DeviceNet/ControlNet/EtherNet IP Fieldbus and 7.8 Interbus Fieldbus.*

Position Monitoring Group

Selecting ***Commission ⇒ Position Monitoring Group*** opens the *Position Monitoring Group* window. In offline mode, the Position Group Monitoring window is used to configure up to 8 groups. In online mode, the window is used to view the status of current, peak and maximum deviation values for all groups.

For specific information on how to commission PMG, refer to section 5.3 in volume 1.

Rexroth
Bosch Group

PLS (Programmable Limit Switch)

A **P**rogrammable **L**imit **S**witch is used to switch on and off digital outputs based on the input position of an associated axis or master. The parameterization of Control, Drive and Option Card PLSs can be done with the VisualMotion's PLS tool.

To launch the PLS tool, select ***Commission*** ⇒ ***PLS*** from VisualMotion's main menu.

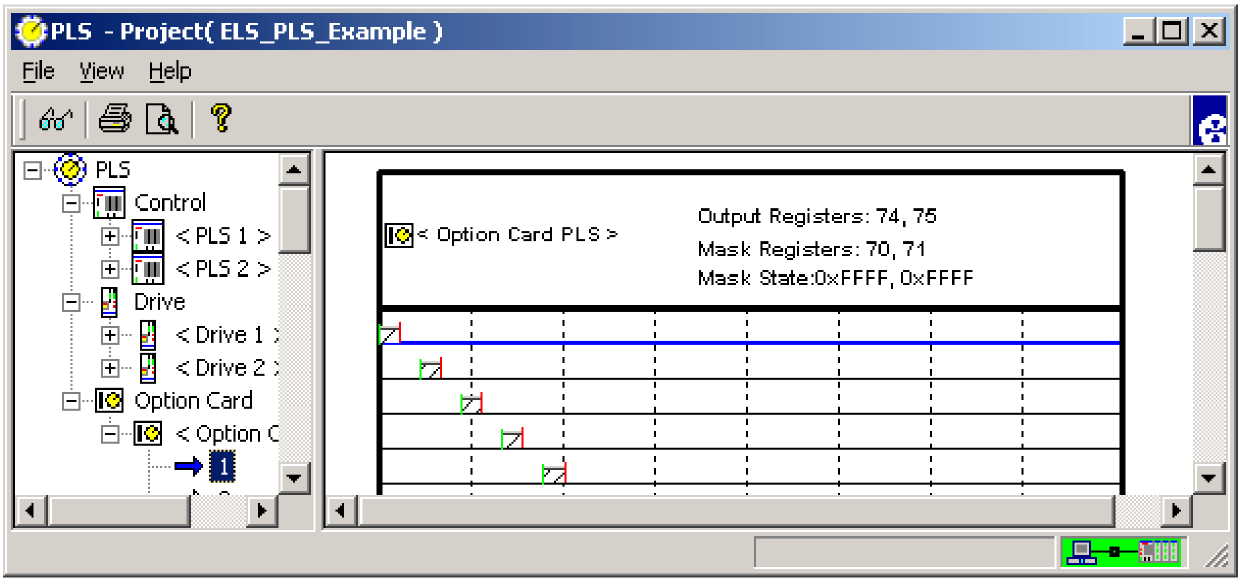


Fig. 13-33: PLS Configuration Tool

For specific information on how to commission a PLS, refer to section 5.5 in volume 1.

Coordinated Motion

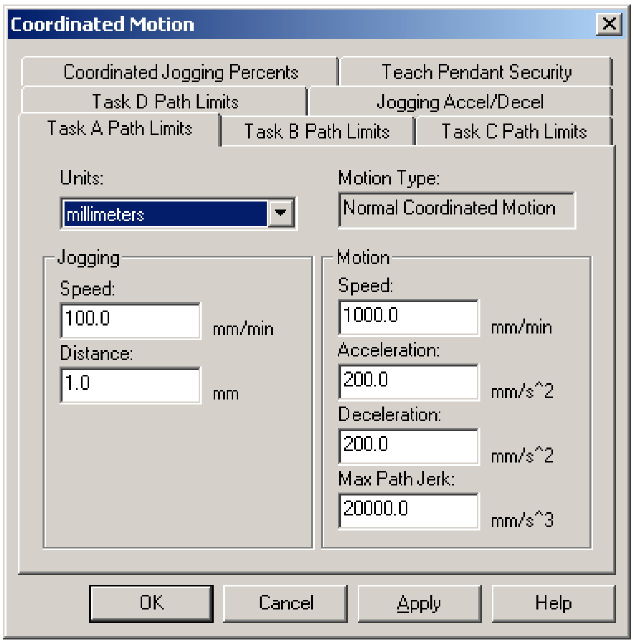
The Coordinated Motion menu selection is available in online or service modes. The standard tabs that are normally displayed are as follows:

- Task Path Limit
- Jogging Accel/Decel
- Coordinated Jogging Percents
- Teach Pendant Security

Task (A-D) Path Limits

The Task Maximum Path Limits window allows speed, acceleration and deceleration limits for coordinated motion to be individually set for each task.

Note: The Task (A-D) Path Limit tabs are displayed only if an axis is configured and associated to a task. The association of axis to task is done under ***Setup*** ⇒ ***Axes***.



Commission_Menu.tif

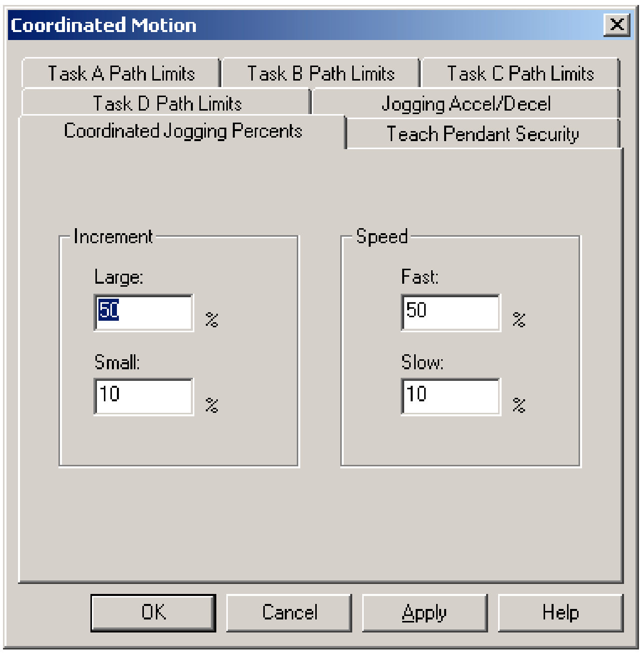
Fig. 13-34: Commission ⇒ Coordinated Motion

This window allows setting the units (inches, millimeters or radians) for the Path Limit values. It also permits setting the jogging speed and distance, setting the motion speed, and assigning separate values for acceleration and deceleration. The drive's internal function uses the same acceleration/deceleration value for single-axis non-coordinated motion. The user can also set the Max Path Jerk for the task.

Clicking **OK** downloads the changed values to the control, without requiring the control to be in parameter mode.

Coordinated Jogging Percentage

This window allows the user to set the increments and velocities for fast and slow jogging for all configured coordinated axes.



Commission_Coord_Jog.tif

Fig. 13-35: "Coordinated Jogging Percents" Window

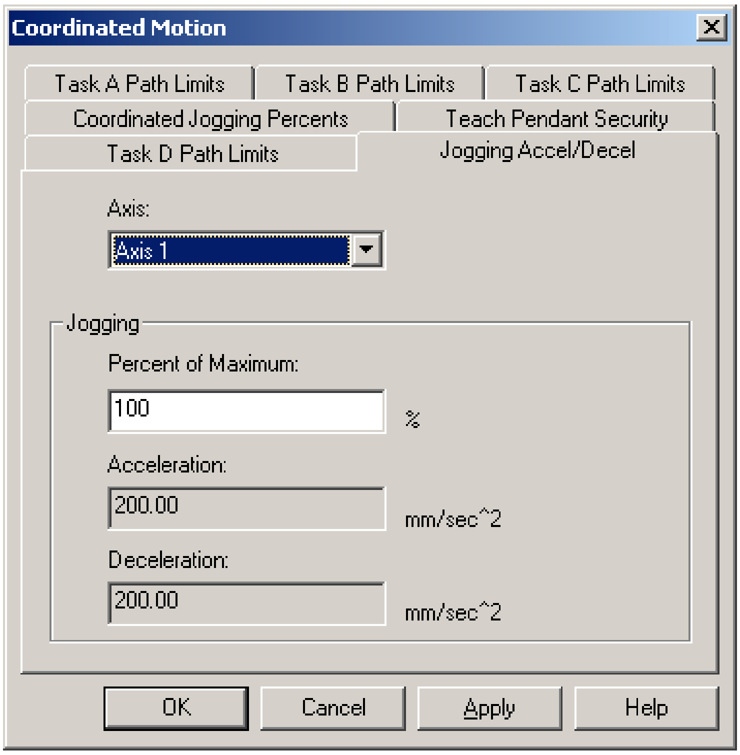
Rexroth
Bosch Group

The **Increment** data area is used to set the **Large** and **Small** percentage of the maximum distance for a single-step jog operation. The maximum is defined by the axis parameter "Maximum Jog Increment" (T-0-0025). Similarly, the **Speed** data area is used to set the Fast and Slow jog speeds as a percentage of the maximum velocity, which is defined by the axis parameter "Maximum Jog Velocity" (T-0-0026). These values are stored in the following parameters:

- Large Increment (C-0-0052)
- Small Increment (C-0-0053)
- Fast Speed (C-0-0055)
- Slow Speed (C-0-0056)

Jogging Acceleration

This window allows the user to set the jogging acceleration and deceleration for each axis. Refer to Parameter *A-0-0021* and *A-0-0022* for more information.



Commission_Jog_Accel.tif

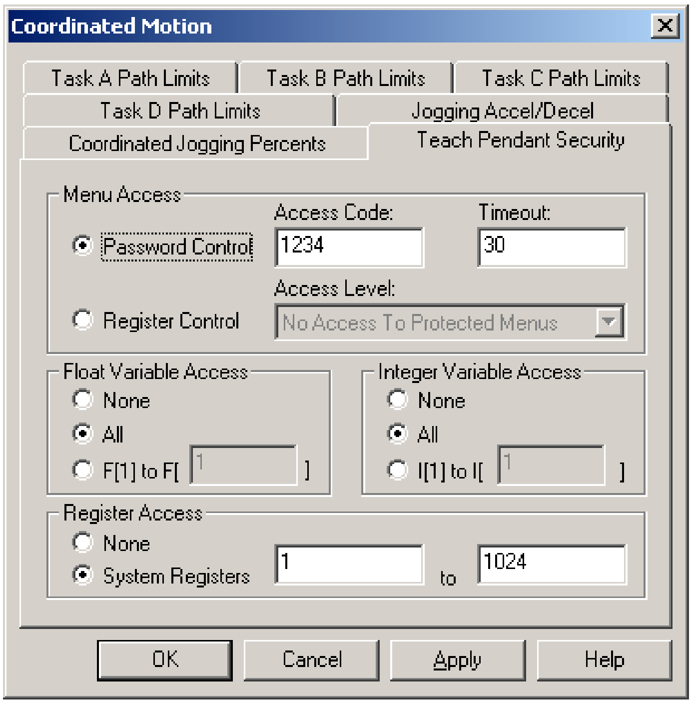
Fig. 13-36: “Jogging Accel/Decel Control” Window

Clicking **OK** downloads the changed values to the control, without requiring the control to be in Parameter Mode.

Pendant Security

This window allows selection of the level of user accessibility for variables and registers when using the BTC06:

- ***Access Code –*** sets the password (access code) that must be entered to gain access to the designated variables and registers.
- ***Password Timeout –*** restricts the time the user has to enter the password (in sec).
- ***Float Variables Access*** – sets user access to floats: None, All, or a range of specified floats.
- ***Integer Variables Access*** – sets user access to integers: None, All, or a range of specified integers
- ***Register Control Access Level -*** No Access to Protected Menus, Basic User Level Menu Access, Supervisor Level Menu Access, Full Menu Access
- ***Register Control –*** Enables Register Control Access Level.



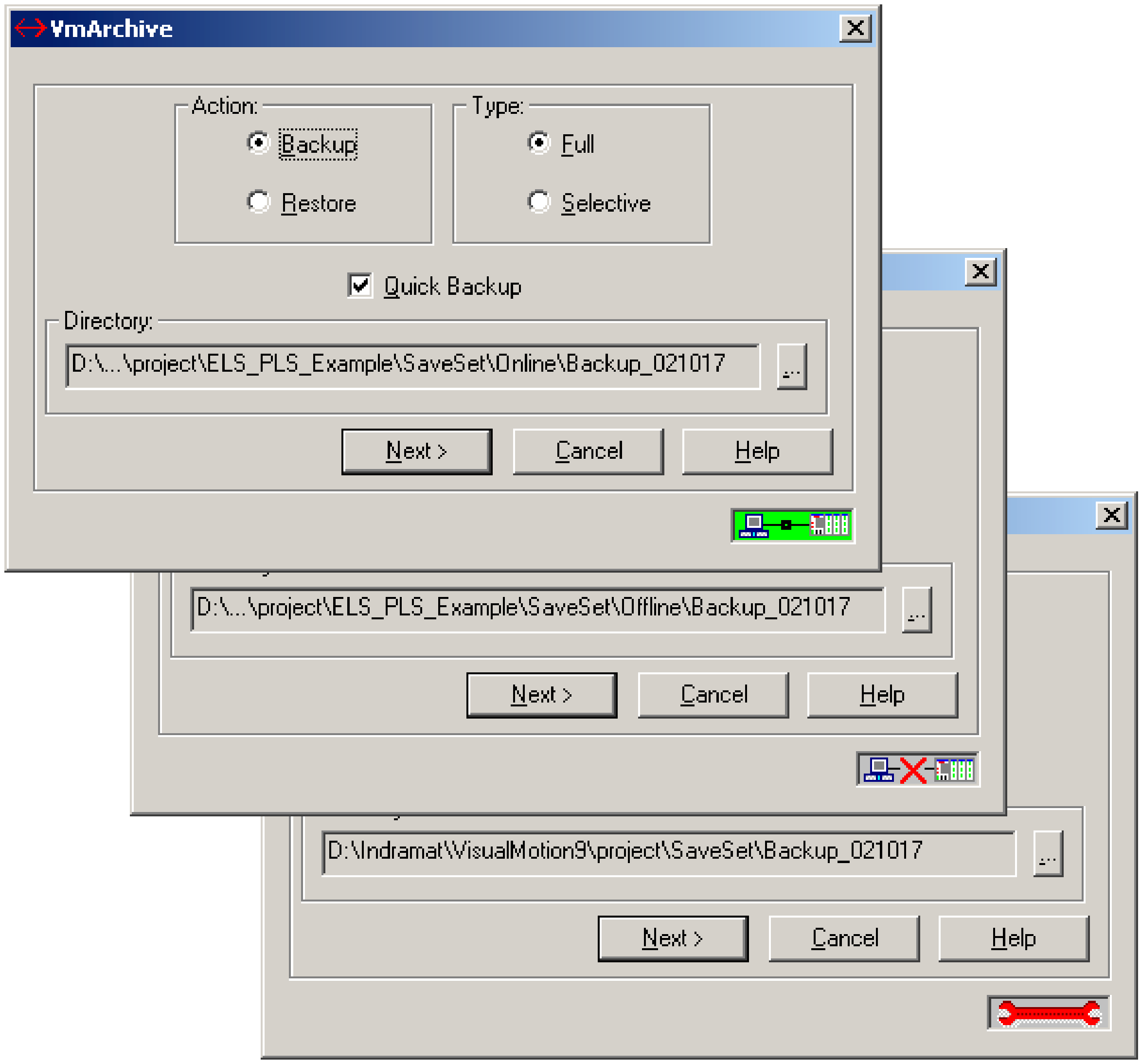
Commission_Tch_Pndnt.tif

Fig. 13-37: “Teach Pendant Security” Window

Archive

Selecting ***Commission*** ⇒ ***Archive*** ⇒ ***System…*** or clicking the Archive icon opens the *VmArchive* window in Fig. 13-38.

These windows provide backup and restore functionality of project data in online, offline or service modes.



Commission_Archive.tif

Fig. 13-38: “Archive” Window

For specific information on how to archive a project, refer to section 10.2 in volume 1.

Archive History and Flash Backup

Selecting ***Commission*** ⇒ ***Archive*** ⇒ ***Other …*** opens the *Archive Other* window. The tabs in this window allow the user to retrieve information about existing archives and to perform backups for the Non-volatile and compact flash memory areas.

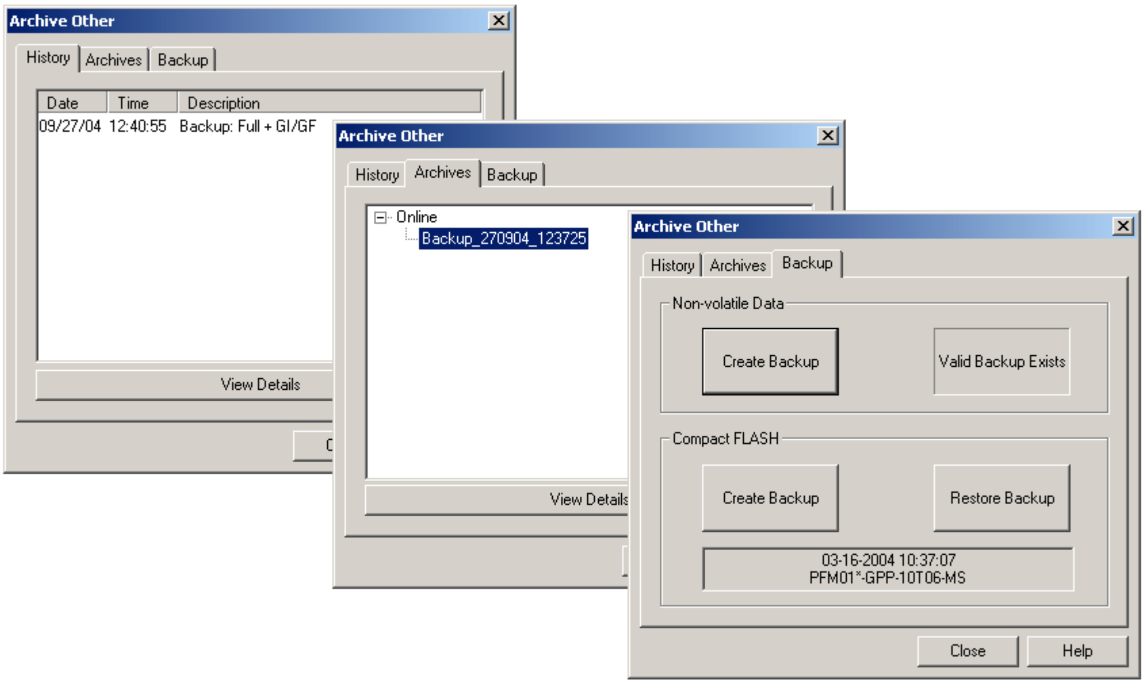


Fig. 13-39: Archive Other

For specific information on how to backup to NVRam and compact flash, refer to section 10.3 in volume 1.

Transfer

The transfer utility is used to import and export data for the program currently opened while in service mode.

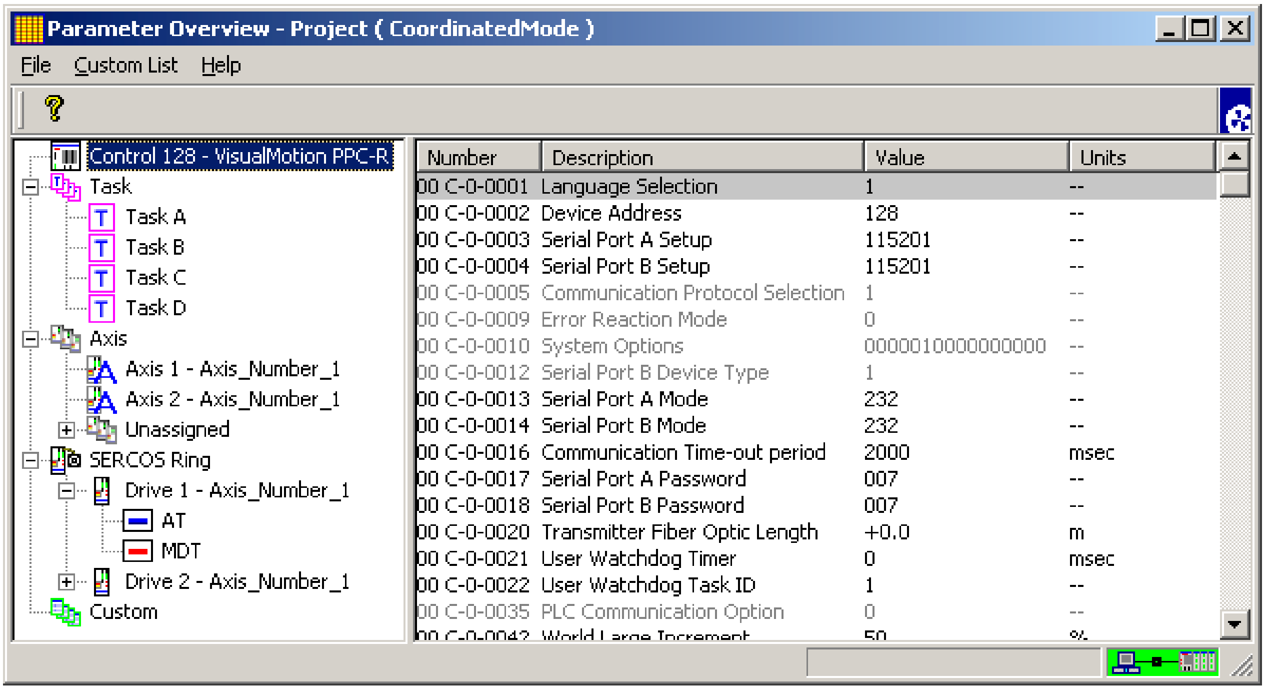
For specific information on how to use Transfer, refer to section 10.7 in volume 1.

13.8 The Data Menu

The **Data** menu provides the user access to VisualMotion system parameters, registers, variables, events, points and zones. In addition to system data, the user can access runtime utilities such as CAM Indexer, ELS, PID and Registration.

Parameters

Selecting ***Data*** ⇒ ***Parameters*** ⇒ ***Edit*** opens the from *Parameter Overview Runtime Tool* window.



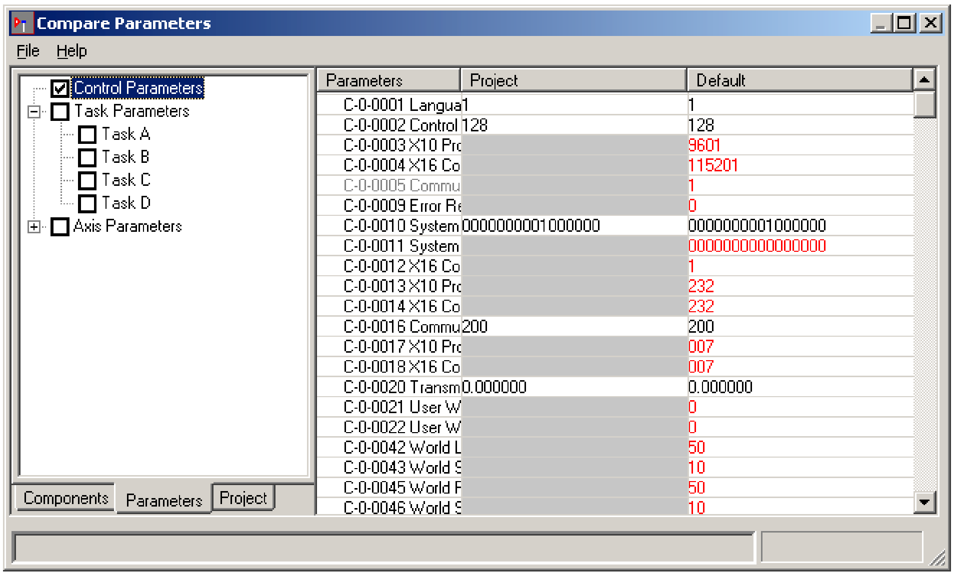
Data_Param.tif

Fig. 13-40: Parameter Overview Window

For specific information on how to use the parameter overview, refer to section 8.1 in volume 1.

Compare Parameters

Selecting ***Data*** ⇒ ***Parameters*** ⇒ ***Compare*** opens the from *Compare Parameters* window.



Compare_Param.tif

For specific information on how to use compare parameters, refer to section 8.1 in volume 1.

Registers

Selecting ***Data*** ⇒ ***Registers*** opens the *Data Editor Tool* window displaying all 1024 (16-bit) registers available on the control.

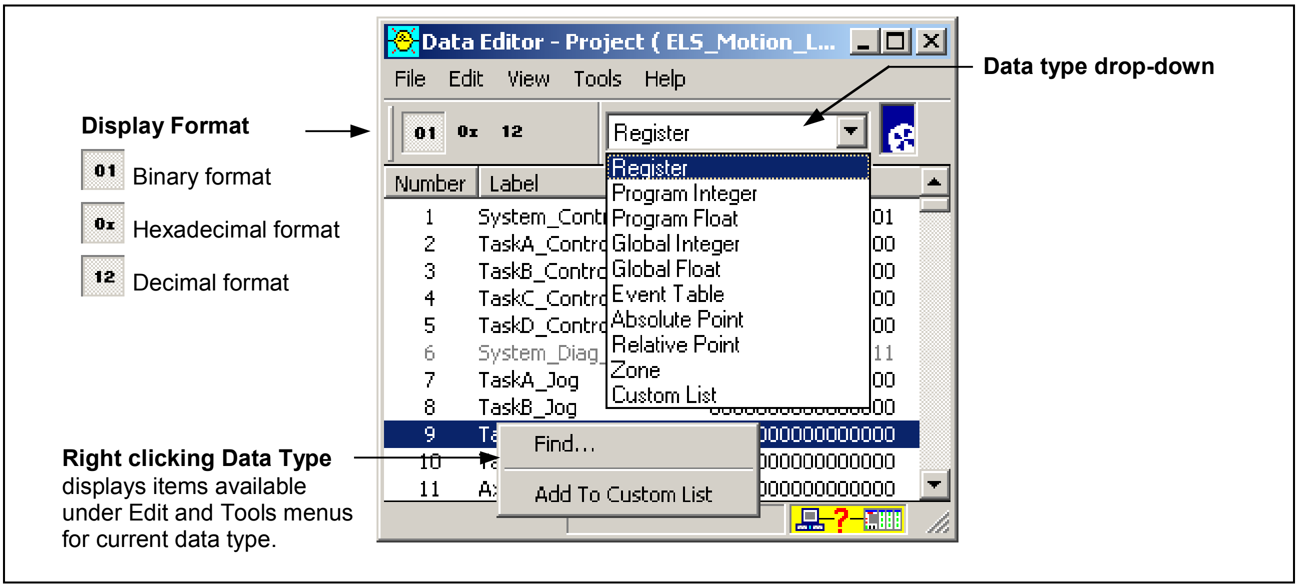


Fig. 13-41 Data Editor displaying Registers

For specific information on how to display and edit registers, refer to section 8.2 in volume 1.

Variables

Selecting ***Data*** ⇒ ***Variables*** opens the *Data Editor Tool* window and automatically uploads and displays variables in the active project on the control.

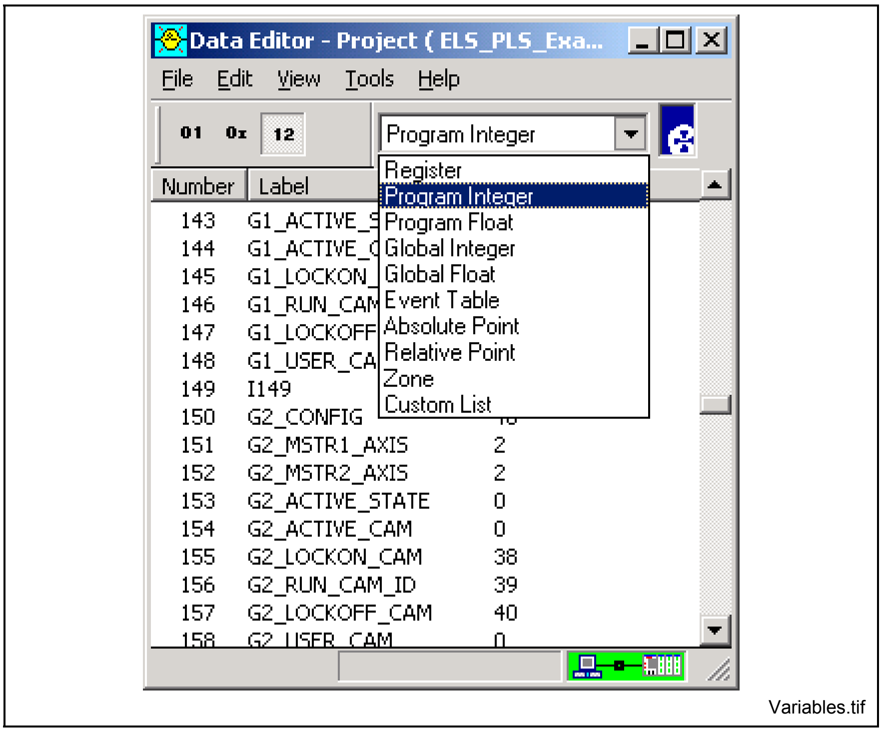
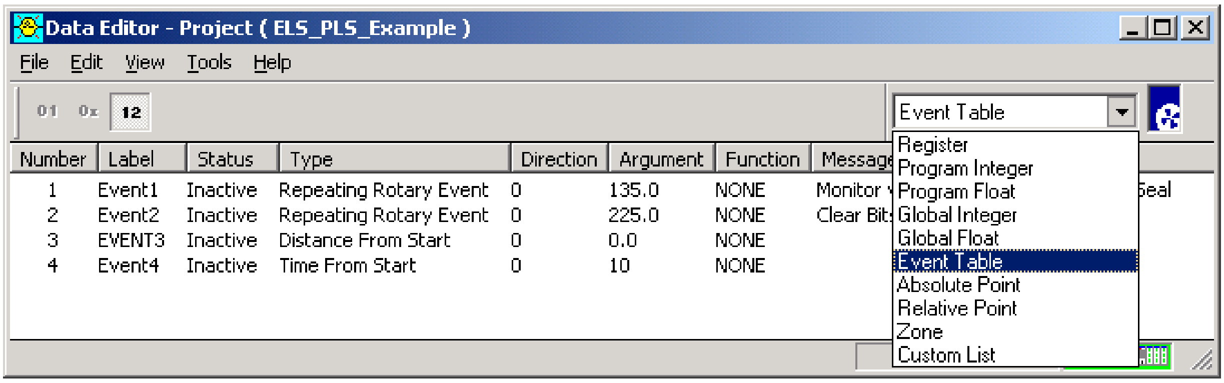


Fig. 13-42: Data Editor displaying Variables

For specific information on how to display and edit program and global variables, refer to section 8.2 in volume 1.

Event Tables

Selecting ***Data*** ⇒ ***Events*** opens the *Events Runtime Tool* window.



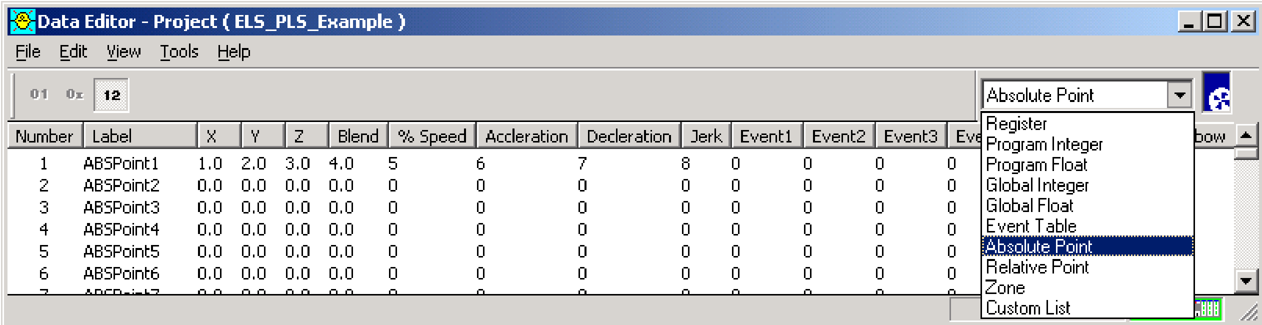
Event_List.tif

Fig. 13-43: Data Editor displaying Event Tables

For specific information on how to display and edit event tables variables, refer to section 8.2 in volume 1.

Points

Selecting ***Data*** ⇒ ***Points*** opens the *Points Runtime Tool* window.



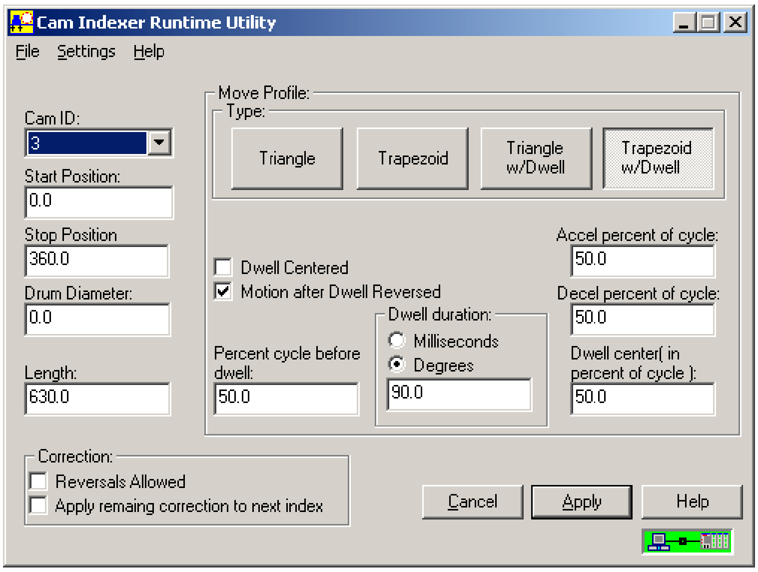
Points.tif

Fig. 13-44: Data Editor displaying Points

For specific information on how to display and edit point tables, refer to section 8.2 in volume 1.

CAM Indexer

Selecting ***Data*** ⇒ ***CAM Indexer*** opens the *CAM Indexer Runtime Tool* window.



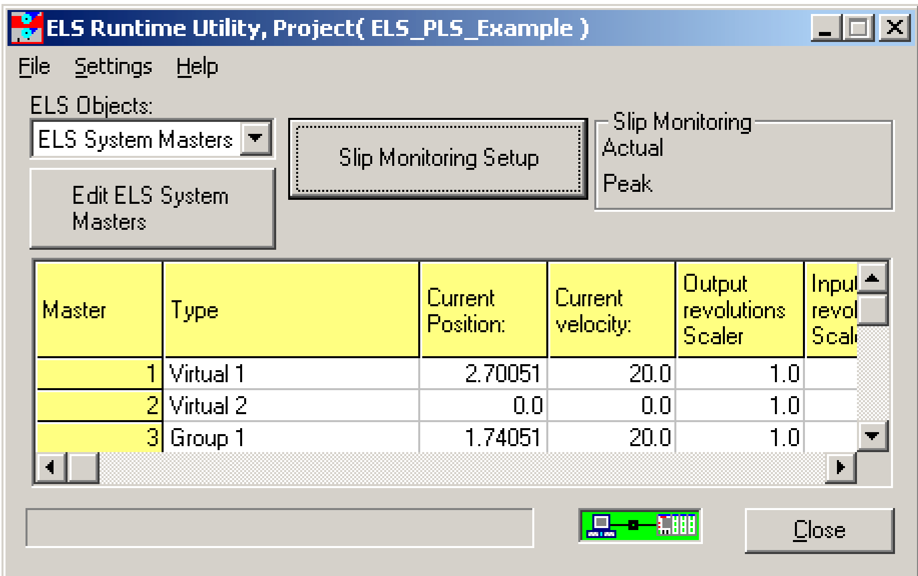
CamIndexer.tif

Fig. 13-45: CAM Indexer Runtime Utility

For specific information on how to display and edit a CAM Indexer, refer to section 8.4 in volume 1.

ELS

Selecting ***Data*** ⇒ ***ELS*** opens the *ELS Runtime Tool* window below.



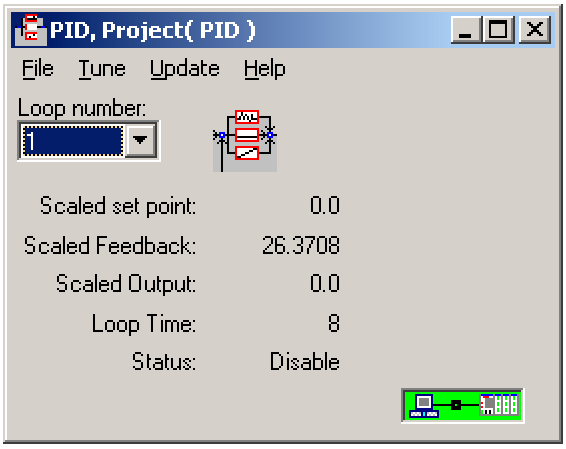
ELS_Utility.tif

Fig. 13-46: ELS Runtime Utility

For specific information on how to use the ELS Runtime Tool, refer to section 8.3 in volume 1.

PID

Selecting ***Data*** ⇒ ***PID*** opens the *PID Monitor Runtime Tool* window. This tool is used for monitoring and tuning the PID of the active program on the control. PID's are added to a program under ***Setup*** ⇒ ***Processes*** ⇒ ***PID Loops***.



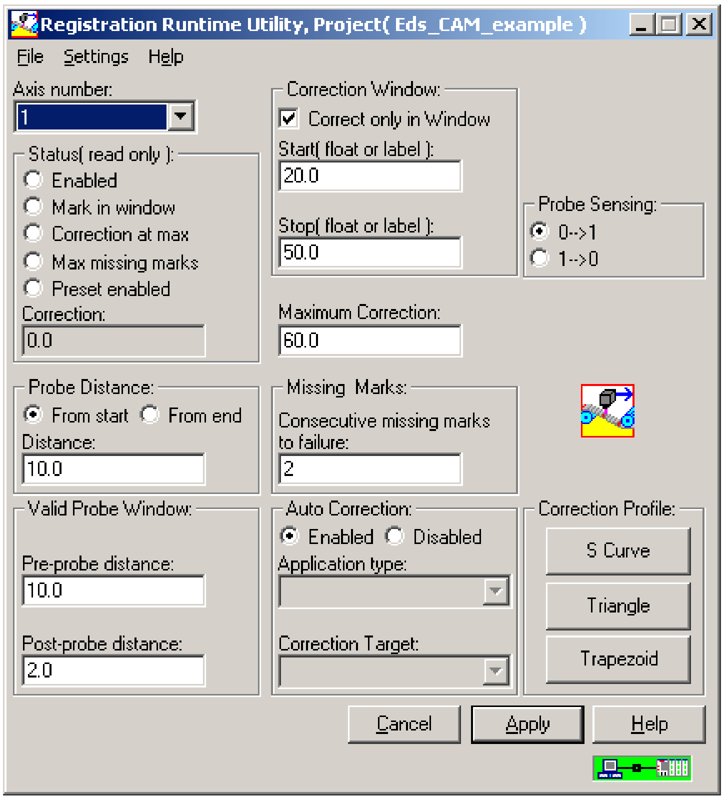
PID.tif

Fig. 13-47: PID Monitor Window

For specific information on how to use the PID Tool, refer to section 5.4 in volume 1.

Registration

Selecting ***Data*** ⇒ ***Registration…*** opens the *Registration Runtime Tool* window.



Registration.tif

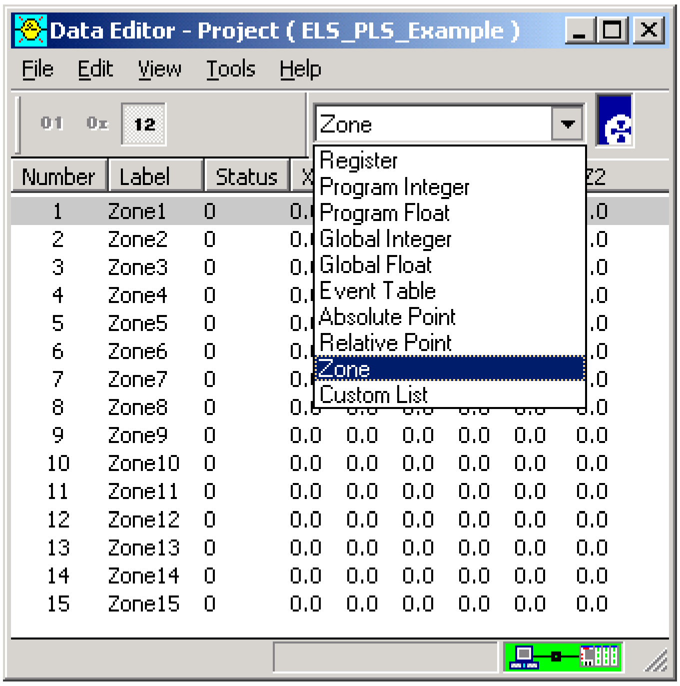
Fig. 13-48: Registration Utility

Note:	Registration is configured in the *Index* icon by checking the ***Enable Registration Correction*** checkbox and then clicking on the ***Registration Correction Setup…*** button.

For specific information on how to use Registration, refer to section 8.4 in volume 1.

Zones

Selecting ***Data*** ⇒ ***Zones*** opens the *Zones Runtime Tool* window. This utility allows viewing and editing of the zone table on the control.



Zones.tif

Fig. 13-49: Zones List

For specific information on how to display and edit zones, refer to section 8.2 in volume 1.

13.9 The Diagnostics Menu

The **Diagnostics** menu is used to monitor system information such as system status, drives and tasks. The user can also analyze a signal using the Oscilloscope tool, and program flow by enabling a Breakpoint.

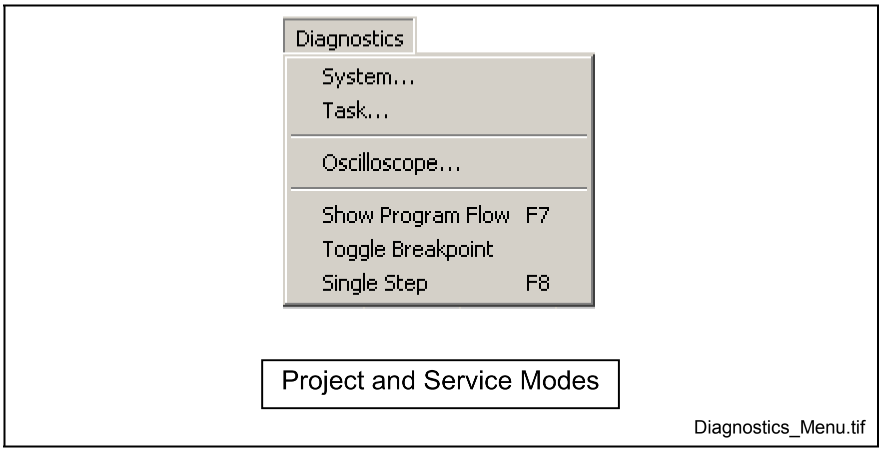


Fig. 13-50: The Diagnostic Menu

System

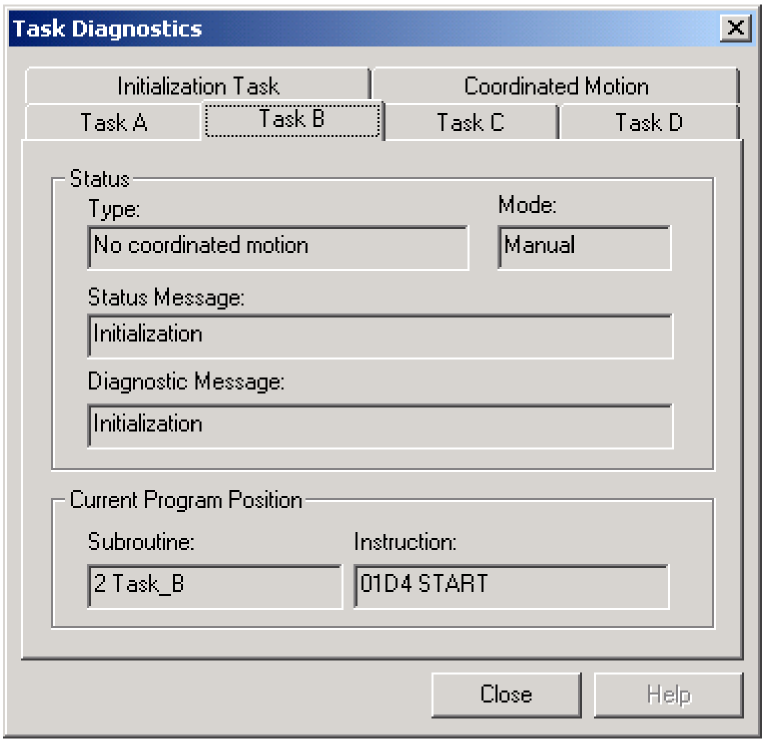
Selecting ***Diagnostics*** ⇒ ***System*** in online mode opens the *System Diagnostics* window. This window displays diagnostic information about the active VisualMotion system. The window uses tabs to display diagnostic information for the following:

- Status
- General
- Option Cards
- Diagnostic Log
- IndraLogic Log
- Hardware
- Load (System)
- Integrated PLC
- MEC Status

For specific information on system diagnostics, refer to section 9.4 in volume 1.

Tasks

Selecting ***Diagnostics*** ⇒ ***Task*** opens the *Task Diagnostics* window and uploads data regarding all active VisualMotion tasks. Task letters are displayed only if they contain an icon program that has been compiled and downloaded to the control. All GPP 11 and GMP 11 programs will contain the Initialization Task and Task A tabs. The Coordinated Motion tab is only visible if any task contains a coordinated program. It displays the active coordinated axes and their current X, Y, Z positions.



Tasks.tif

Fig. 13-51: Tasks

Status - indicates the type of motion programmed in the selected task for the active program and the current control mode (Parameter, Initialization, Manual or Automatic). Status and Diagnostic messages for the selected task are also displayed.

Current Program Position - displays the subroutine and instruction executing and its pointer. This display is useful when debugging in single-step mode. If a program is running in automatic mode, the displayed instruction is the instruction that was executing at the time that the operating system sampled the instruction execution.

Drive Diagnostics

Selecting ***Diagnostics*** ⇒ ***Drive Diagnostics*** opens the *Drive Diagnostics* window and uploads the following read-only drive information for all digital drives found on the Sercos drive ring:

- Current Diagnostics Message
- Version Information:
  - Installed drive firmware
  - Hardware (drive type)
  - Motor type
  - Control section type (IndraDrive) and Serial # (EcoDrive or Diax)

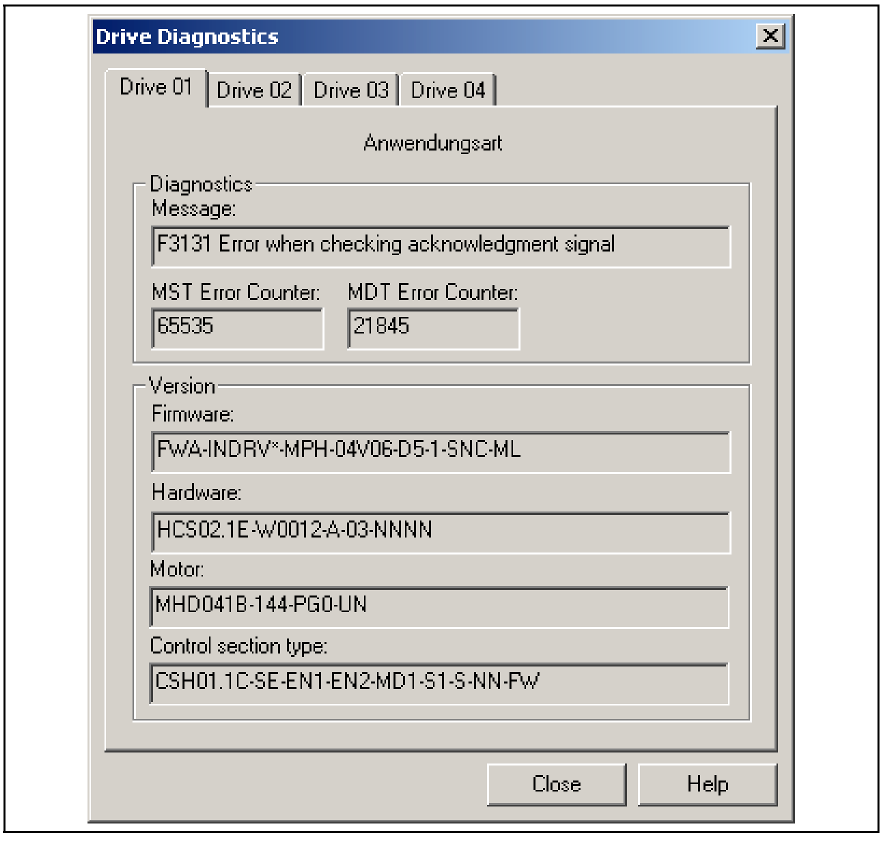
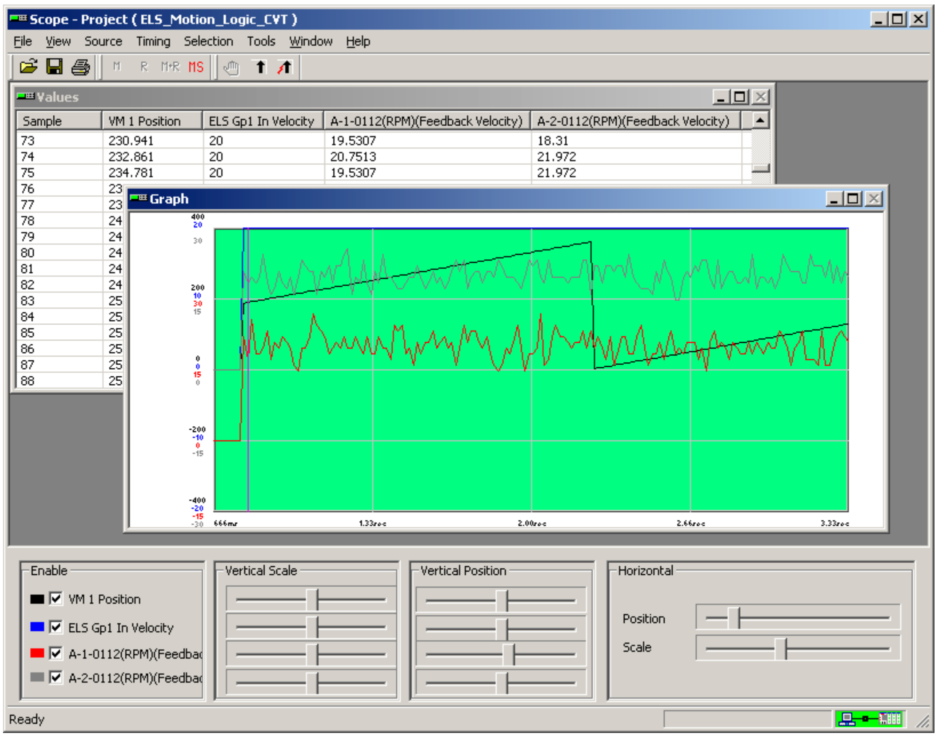


Fig. 13-52: Drive Diagnostics

Oscilloscope

VisualMotion's oscilloscope function is used to capture predefined internal and external signals from the control or connected drive(s).



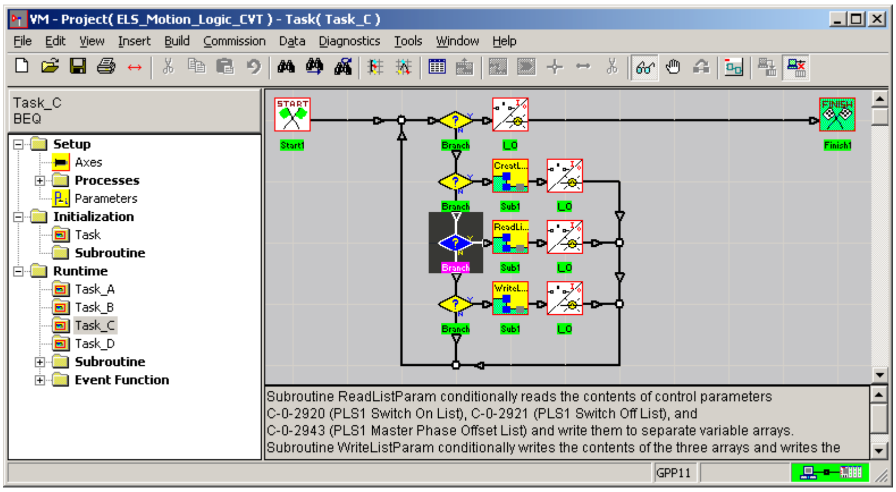
oscillo_utility.tif

Fig. 13-53: Oscilloscope Utility

For specific information on how to use the Oscilloscope, refer to section 9.1 in volume 1.

Show Program Flow F7

Selecting ***Diagnostics*** ⇒ ***Show Program Flow F7*** highlights the currently executing icon in a running program, as illustrated in Fig. 13-54.



Program_Flow.tif

Fig. 13-54: Show Program Flow View

For specific information on Show Program Flow, refer to section 9.2 in volume 1.

Toggle Breakpoint

Selecting ***Diagnostics*** ⇒ ***Toggle Breakpoint*** sets a breakpoint in the currently displayed task or subroutine. This menu selection is enabled only after the ***Show Program Flow*** function is selected. Breakpoints can only be set for task A-D and subroutines. The initialization task, initialization subroutines and event functions do not support breakpoints.

When the program is executed, program flow stops on the first icon after the Start icon in the selected function (task or subroutine).

Note: This function is only available in online or service mode.

For specific information on how to use Breakpoint, refer to section 9.3 in volume 1.

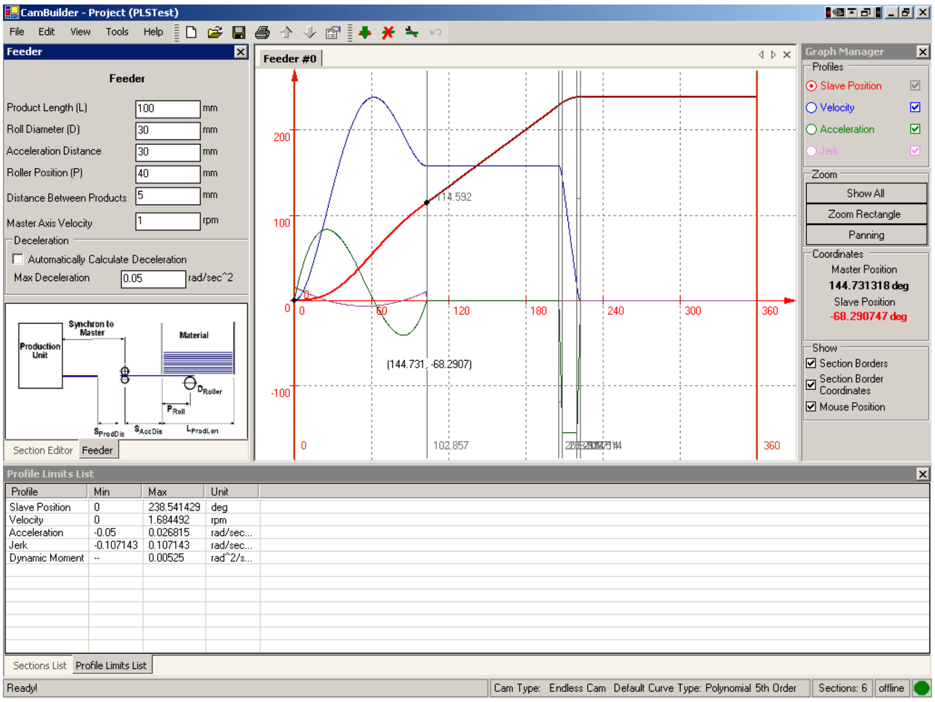
13.10 The Tools Menu

The **Tools** menu contains the CAM Builder and Jogging utilities. The user can also launch additional Rexroth Registered Tools. Items such as different releases of the DDE Server, IoBox and even other instances of VisualMotion Toolkit can be launched. Communication settings and language selection are also available from the Tools menu.

CamBuilder

Selecting ***Tools*** ⇒ ***CamBuilder*** opens the CAM building tool that was installed during the initial installation of VisualMotion 11. Refer to section 2.3, *Installation*, in volume 1 for details on optional installation of the CamBuilder tool.

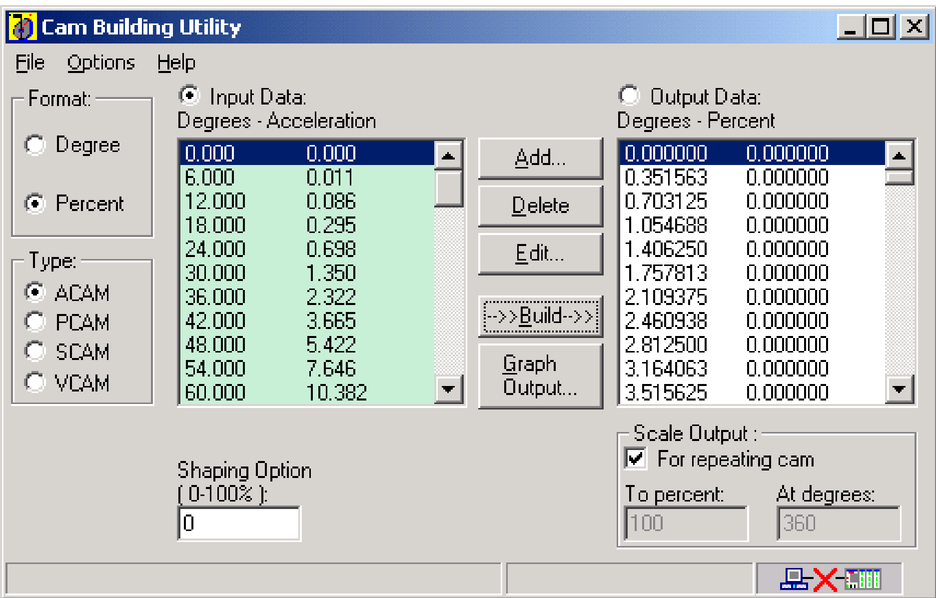
This tool is used to build CAMs for Rexroth's motion controls and drives. The following figure shows the new CAM Builder. Documentation for the New CamBuilder is provided only as an online help system.



CAM_Builder.tif

Fig. 13-55: New CAM Builder Tool

The following figure shows the old CAM Builder tool used with VisualMotion 9. Refer to section 8.5, *CAM Builder*, in volume 1 for details.



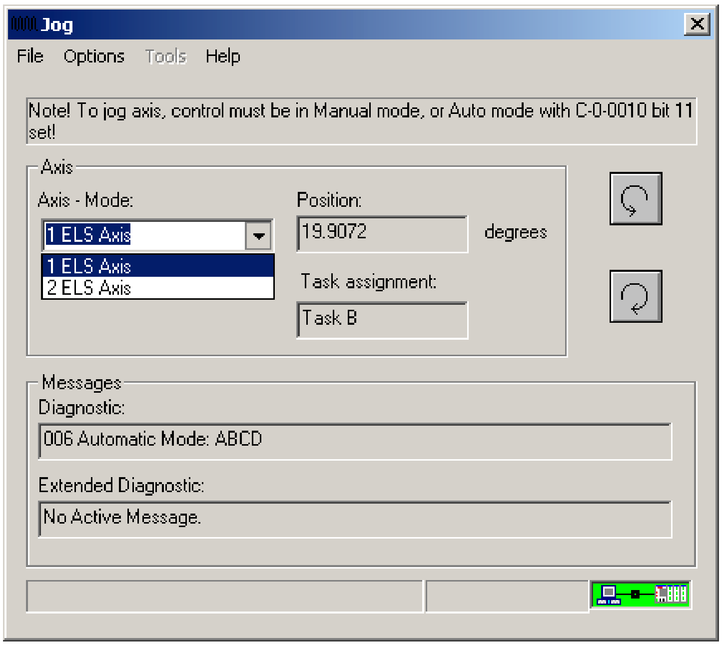
CAM_Builder.tif

Fig. 13-56: Old CAM Builder Tool

Jogging

Selecting ***Tools*** ⇒ ***Jogging*** opens the *Jog Tool* window. This tool is used to jog an axis in manual or automatic mode. Only axes that configured or switched to either *Velocity* or *Single Axis* operating modes can be jogged using this tool.

Note: This tool is only available in online and service modes.



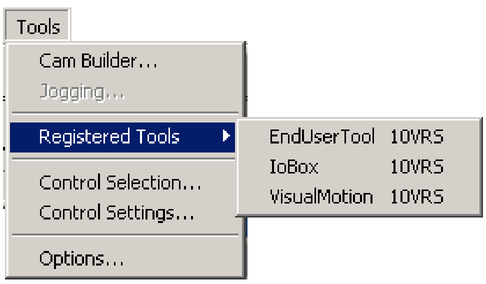
Jogging.tif

Fig. 13-57: Jog Window

For specific information on how to use Jogging, refer to section 7.5 in volume 1.

Registered Tools

The menu selection ***Tools*** ⇒ ***Registered Tools*** displays by default VisualMotion tools that were installed during the initial installation of the software. Any executable program file can be added to this list by adding the exe file under ***Tools*** ⇒ ***Options*** ⇒ ***Tools.***



RegisteredTools.tif

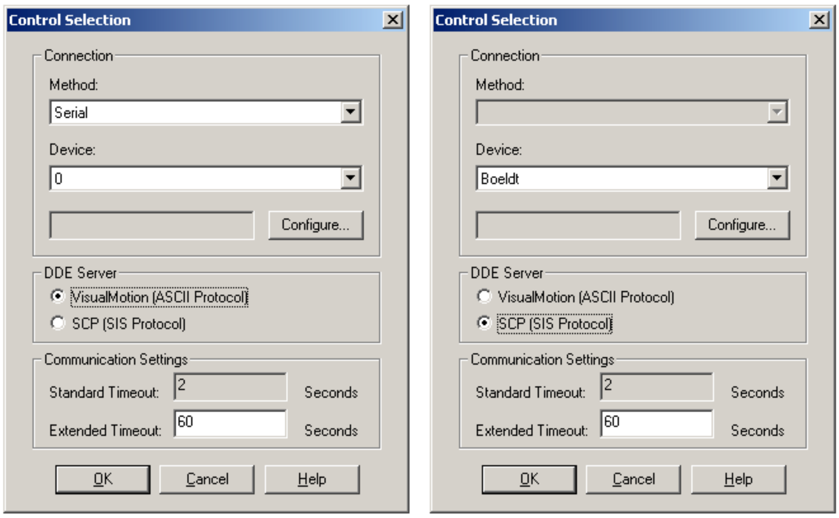
Fig. 13-58: Registered Tools

Refer to Tools Options on page 13-64 for details.

Control Selection

Selecting ***Tools*** ⇒ ***Control Selection*** in service or offline mode opens the *Control Selection* window. This window is used to select the *Connection* type and *Dde Server* that will be used to communicate with the motion portion of the control.

The options available in the *Control Selection* window are dependent on the selected Dde Server.



ControlSelect.tif

Fig. 13-59: Card Selection Setup Window

Method (Connection)

This field is available only when using the VisualMotion (ASCII Protocol) server. The methods available are either **Serial** or **EtherNet**.

Target

When using the VisualMotion (ASCII Protocol) server, this field identifies the control's address in the Sercos ring for a serial connection and the device name for a Network connection.

When using the SCP (SIS Protocol) server, the target represents the device name assigned using the SCP Systemconfigurator.

The ***Configure…*** button opens the VM Dde communication window when using the VisualMotion (ASCII Protocol) server and the SCP Systemconfigurator when using the SCP (SIS Protocol) server.

For specific information on how to use the SCP Systemconfigurator, refer to *section 18.3.* For specific information on ASCII protocol and server windows, refer to the *VisualMotion 9 Application* manual.

Dde Server

The servers available are dependent on the selected communication channel under menu selection ***Tools*** ⇒ ***Options*** ⇒ ***IndraLogic***. Refer to IndraLogic Options on page 13-59 for details.

The **SCP (SIS Protocol)** server is always available for GPP 11 or GMP 11 projects.

The **VisualMotion (ASCII Protocol)** server is only available when using a *Direct RS232 Connection on X16* or *Direct EtherNet Connection for* IndraLogic. The VisualMotion (ASCII Protocol) server is not available when using the *VisualMotion Connection (SCP)* for IndraLogic.

Communication Settings

Standard Timeout is set by VisualMotion and is used to set the timeout period for accessing standard data such as parameter and variable values.

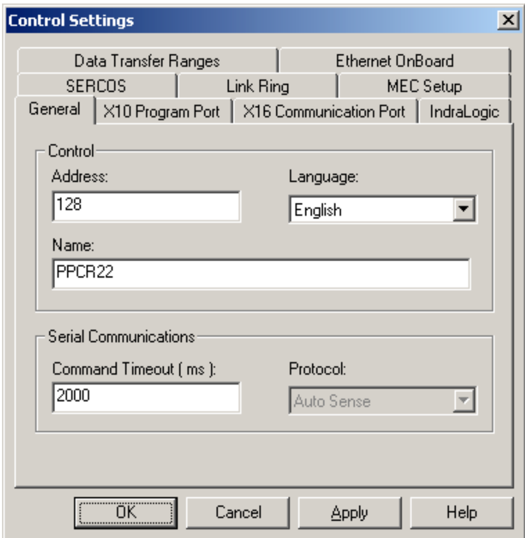
Extended Timeout is defaulted to 60 seconds but can be configured from 1 to 900 (15 min.) seconds and is used to set the timeout period for accessing large amounts of data. Some example would be reading/writing list parameter elements, downloading and activating programs. Typically, 60 seconds is enough time for most communication. However, there may be data processing, such as accessing large IndraDrive PLC programs that may require more than 60 seconds to complete.

Control Settings

The ***Control Settings*** menu selection is used to configure the communication settings between the control and any external interface. In addition to communication settings, the user can also set a password for the X10 Program Port, X16 Communication Port and EtherNet. All fields within the *Communication Settings* window are stored to control parameters. Holding the cursor above any field will display the corresponding parameter number.

General

The General tab is avaliable in project or service mode and is used to set the Control's Address, Communication Timeout and Protocol.



GeneralSettings.tif

Fig. 13-60: General Communication Settings

Setting	Description	Parameter
Control Address:	Sets the control's address	C-0-0002
Language:	Sets the language of the control and all active drives. **Note**: To change the language of a specific drive, use the Parameter Overview tool and modify Sercos parameter S-0-0267.	C-0-0001
Name:	Sets the control's name	C-0-0142
Command Timeout:	Sets the communication time-out period for serial communication. The state of the communication error timer is set to enabled/disabled by start/stop commands from the serial device.	C-0-0016
Protocol:	Sets the current communication protocol recognized by VisualMotion for system communication. Change only in phase 2.	C-0-0005

Table 13-2: General Communication Settings

X10 Program Port

This window is used to modify the X10 serial connection on the control in online or service mode.

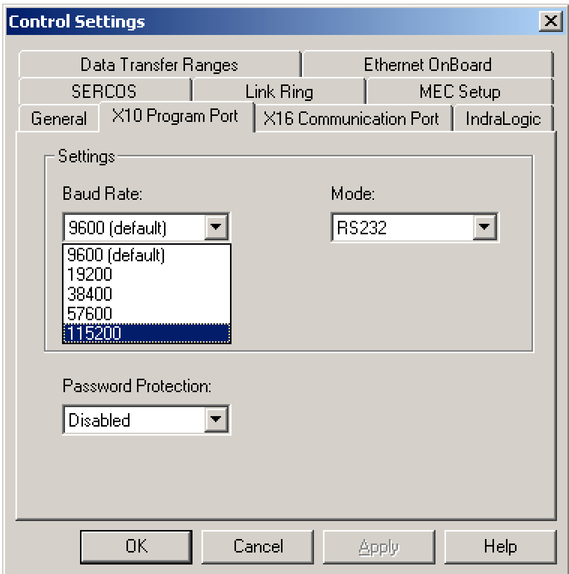


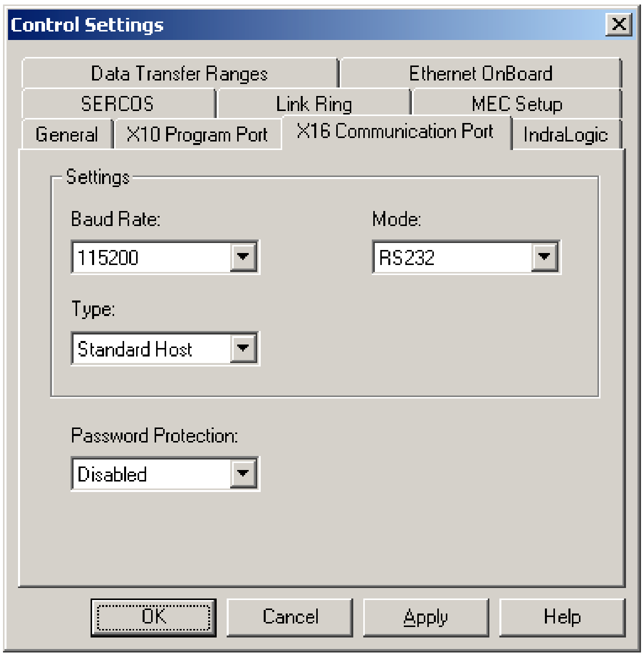
Fig. 13-61: X10 Program Port Settings

Setting	Description	Parameter
Baud rate:	The allowable baud rates for the X10 program port are… 9600 (default), 19200, 38400, 57600 and 115200.	C-0-0003
Mode:	The allowable serial interface modes are RS232, RS422 and RS485.	C-0-0013
Password Protection:	The password protection drop down list allows the user to set a password to limit the access to the X10 port. The allowable settings are Disabled (default), Read/Write and Read Only. Refer to Password Protection on page 13-56 for details.	C-0-0017

Table 13-3: X10 Program Port Settings

X16 Communication Port

This window is used to modify the X16 serial connection on the control in online or service mode. The port is typically used to communicate with a teach pendant (default 9600 baud), if one is installed. If an ASCII "dumb" terminal (e.g. a BTC HMI) is used to communicate with a control, the checksum should be disabled.



X16_Port.tif

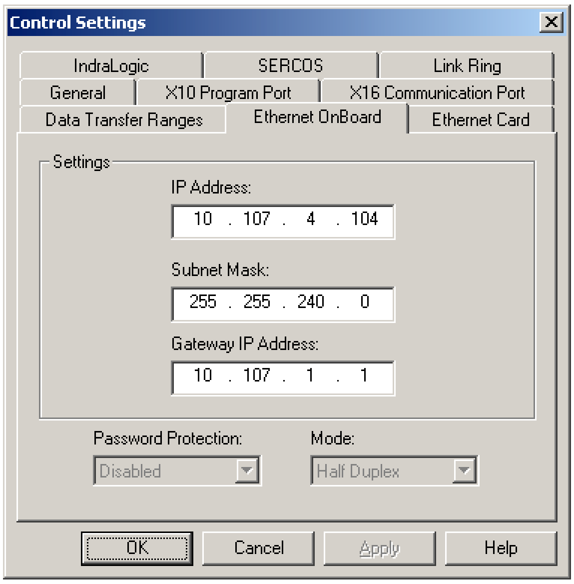
Fig. 13-62: X16 Program Port Settings

Setting	Description	Parameter
Baud rate:	The allowable baud rates for the X10 program port are… 9600 (default), 19200, 38400, 57600 and 115200.	C-0-0004
Mode:	The allowable serial interface modes are RS232, RS422 and RS485.	C-0-0014
Type:	Select Standard host for a PC and Teach Pendant for an HMI. IndraLogic and SysLibCom can also be selected.	C-0-0012
Password Protection:	The password protection drop down list allows the user to set a password to limit the access to the X10 port. The allowable settings are Disabled (default), Read/Write and Read Only. Refer to Password Protection on page 13-56 for details.	C-0-0018

Table 13-4: X16 Program Port Settings

Ethernet OnBoard and Ethernet Card

This window is used to configure the Ethernet settings in online or service mode. The user can set a password to limit access to the control via the network and set the mode of transmission.



network_settings.tif

Fig. 13-63: Network Settings

Note: When both Ethernet interfaces are installed in a PPC-R22.1, the Ethernet option card uses parameter set C-0-0400 through C-0-0405. The onboard Ethernet interface uses set C-0-0411 through C-0-0416. If only one interface is installed, then set C-0-0400 through C-0-0405 is used.

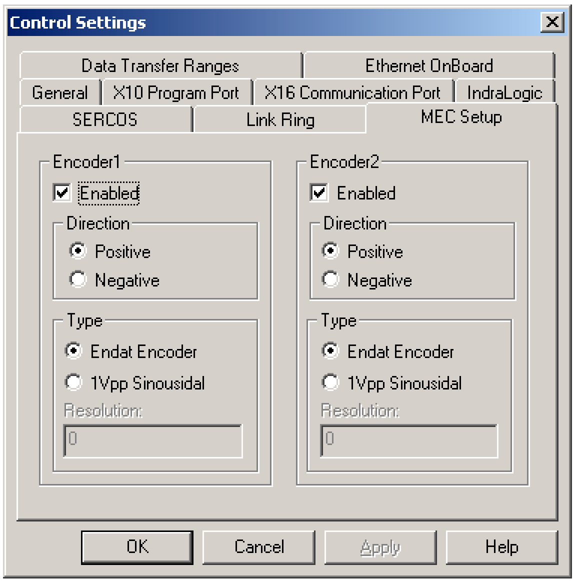
Setting	Description	Parameter	Parameter
Settings:	The EtherNet's IP Address, Gateway IP Address and Subnet Mask are provided to the user by their respective IT department. Every EtherNet must have a unique IP Address assigned.	C-0-0400 C-0-0401 C-0-0402	C-0-0411 C-0-0412 C-0-0413
Mode:	Half Duplex (default) allows transmission in only one direct at a time (receive or transmit). Full Duplex allows bi-directional transmission to and from the control. Full Duplex requires a connection via a LAN switch that supports manual setting of the Duplex mode. The EtherNet does support auto-negotiation, so this specific switch setting is not necessary for full duplex support.	C-0-0403	not applicable
Password Protection:	The password protection drop down list allows the user to set a password to limit the access to the EtherNet. The allowable settings are Disabled (default), Read/Write, Read Only and No Access. Refer to Password Protection on page 13-56 for details.	C-0-0404 C-0-0405	C-0-0415 C-0-0416

Table 13-5: EtherNet Settings

Note: Cycle power to the control for EtherNet settings to take effect.

MEC Setup

This window is used to enable and set the direction and type of encoder(s) that will be used with the MEC master encoder card.



MECsettings.tif

Fig. 13-64: MEC Card Settings

Setting	Description	Parameter
Enabled:	When checked, the corresponding bit 1 and 2 are set to 0.	C-0-2802, Bit 1 & 2
Direction:	sets bit 4 to 0 (positive) or negative (1)	C-0-2815, Bit 4
Type:	When Endat Encoder is selected, the resolution (lines per rev) is automatically detected and used. When 1Vpp is selected, the user sets this value to match the lines per rev of the attached encoder.	C-0-2817 C-0-2818

Table 13-6: MEC Setup Settings

Data Transfer Ranges

This window is used to specify the exact range of program floats and integers to transfer to the control when synchronizing or when transferring variables between programs on the control.

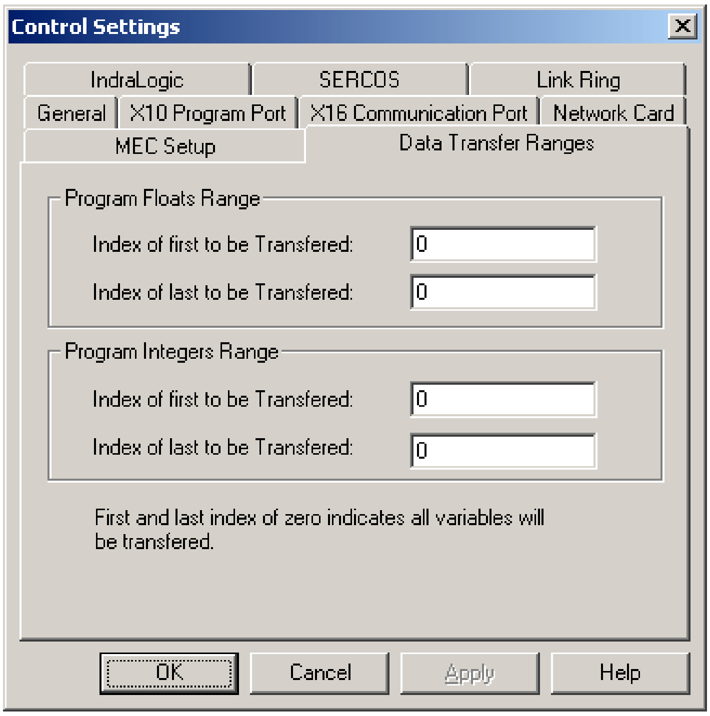
The range values in this window are used by the following transfer utilities:

Program Management

When more than one program exists in the control, the user can transfer variables between two programs on the control. Refer to Data Transfer on page 13-21 for details.

Synchronization

When switching to online mode, the user can choose to transfer the icon program floats and integers to the control. Refer to Project Options on page 13-58 for details.



DataTransRange.tif

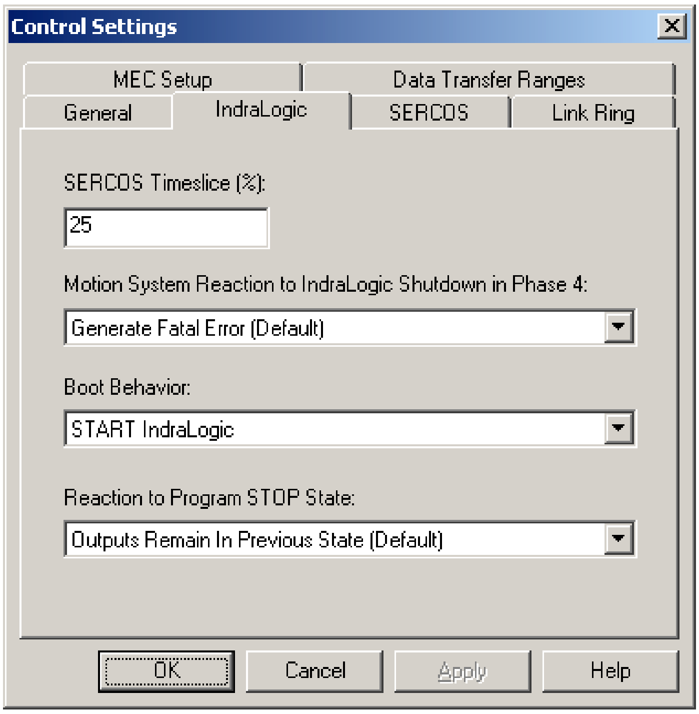
Fig. 13-65: Data Transfer Range

Setting	Description	Parameter
Program Float Range:	sets the starting and ending values for the program floats that can be transferred	C-0-0070 C-0-0071
Program Integer Range:	sets the starting and ending values for the program integers that can be transferred	C-0-0072 C-0-0073

Table 13-7: Data Transfer Range Settings

IndraLogic

This window is used to set IndraLogic timing and reaction settings.



PLC_setting.tif

Fig. 13-66: Integrated PLC Communication Settings

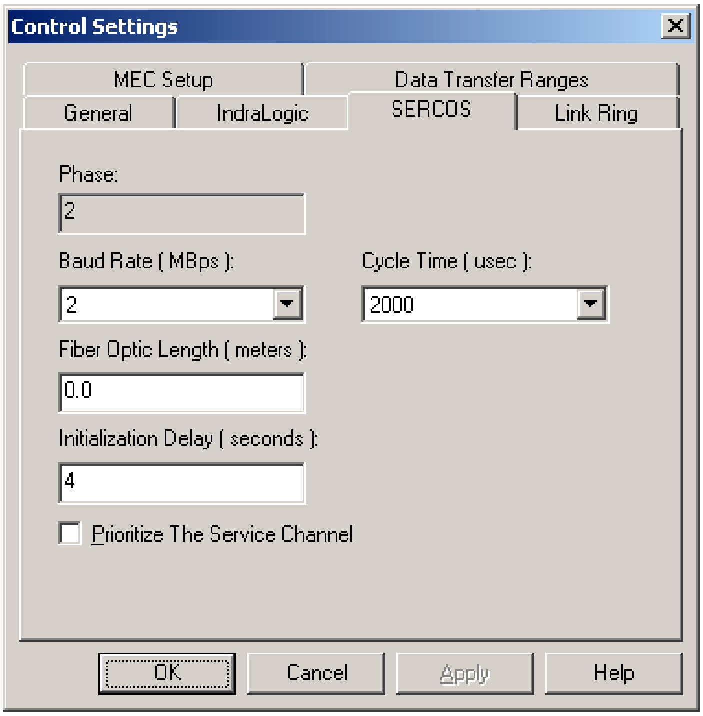
Setting	Description	Parameter
Sercos Timeslice:	This field sets the allocated time for the integrated PLC in the Sercos cycle time.	C-0-1600
Motion System Reaction:	This field configures the PLC error reaction to either generate an error or ignore it.	C-0-1601, Bit 8
Boot Behavior:	This field sets the PLC boot behavior that is used when the PLC project starts.	C-0-1601, Bits 1 & 2
Reaction to Program STOP State:	This field sets the I/O Image output states to remain in their previous state or to reset to 0 when the PLC is stopped.	C-0-1601, Bit 9

Table 13-8: IndraLogic Communication Settings

Rexroth
Bosch Group

Sercos

This window is used to set the Sercos transmission rate and cycle time. In addition, the user can specify the length of the fiber optic transmission cable.



Sercos_setting.tif

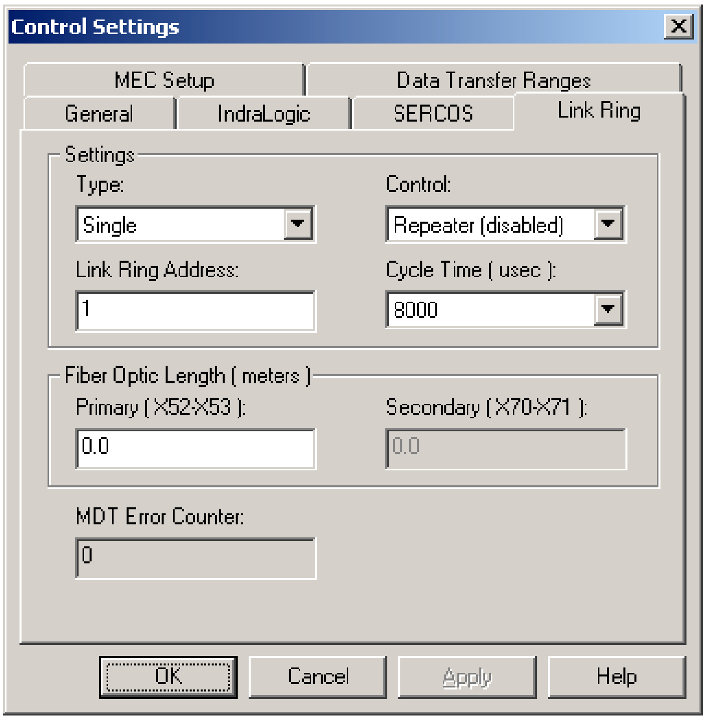
Fig. 13-67: Sercos Settings

Setting	Description	Parameter
Phase:	This read only field displays the current Sercos phase.	C-0-0121
Baud Rate MBps:	Sets the Sercos transmission rate. (Default: 2MBps)	C-0-0010
Cycle Time (usec):	Sets the Sercos cycle time. Refer to Card parameter C-0-0099 for details. (Default: 2000usec)	C-0-0099
Fiber Optic Length:	Sets the intensity of the output from the control's Sercos transmitter, based on the length of the cable in meters. (Default: 0.0)	C-0-0020
Initialization Delay	Causes the control to delay for the specified number of seconds before it initializes the Sercos ring.	C-0-0098

Table 13-9: Sercos Settings

Link Ring

This window is used to set the Link Ring communication settings.



LinkRing.tif

Fig. 13-68: Link Ring Settings

Setting	Description	Parameter
Type:	Sets the Fiber Optic structure as either Single or Double.	C-0-0300
Control:	Sets the control as an active (Master or Slave) participant or a Passive (Repeater) Participant in a Link Ring.	C-0-0300
Link Ring Address:	Up to 32 controls can be used in a Link Ring configuration. The valid range is 1 to 32.	C-0-0304
Cycle Time (usec):	Sets the Link Ring cycle time to 2000, 4000, or 8000 usec.	C-0-0305
Primary:	Sets the output power of the DAQ card to the length of the connected primary fiber optic cable.	C-0-0301
Secondary:	Sets the output power of the DAQ card to the length of the connected secondary fiber optic cable. Used in a Double ring structure.	C-0-0302
MDT Error Counter:	This field displays the illegal master data telegrams (MDT) count by the slave. Entering a 0 resets the counter.	C-0-0303

Table 13-10: Link Ring Settings

For specific information on *Link Ring*, refer to section 6.1 in volume 1.

Password Protection

Password protection can be set for the X10 and X16 serial control ports as well as the EtherNet port. The user can select from either Disabled (default), Read/Write, Read Only and No Access (EtherNet only). From the Password Protection drop-down list (*Tools* ⇒ *Comminations* ⇒ *Control Settings*), select the desired access level and click the ***Apply*** button to open the *Modify Password Setting* window.

Note: Password protection is not supported when using the SCP server.

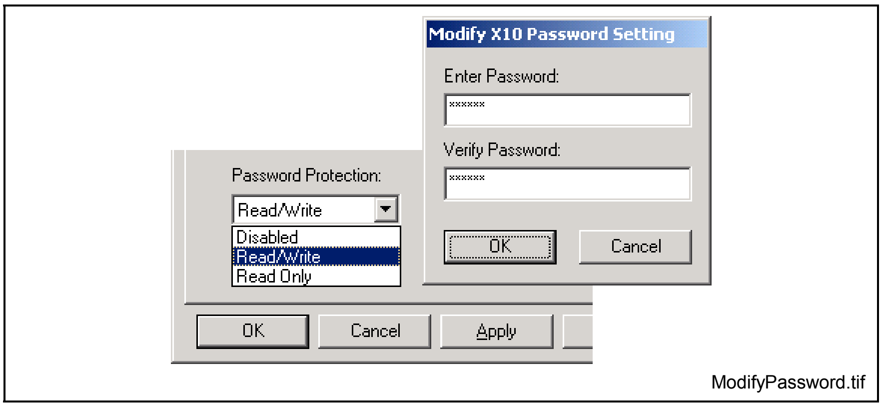


Fig. 13-69: Modify Password Setting

Select a password between 3 to 10 (alpha and/or numeric) characters. The password is not case sensitive and special characters such as "$" or "%" are not allowed.

Changing the Current Password

To change the password, select Disabled and enter the current password. This will clear the current password and allow the user to enter a different password when the process is repeated.

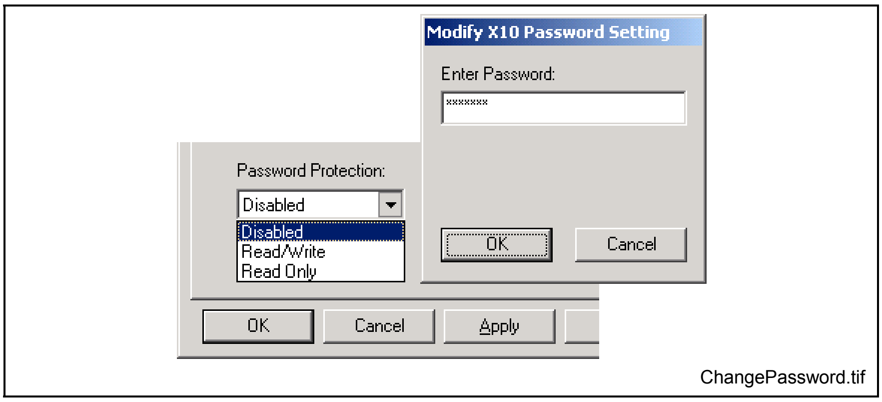


Fig. 13-70: Change Password

Software Restart of Control

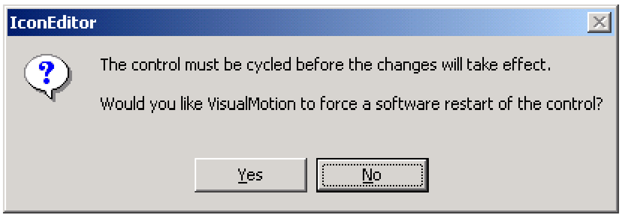
VisualMotion control settings that require a reset for changes to take effect are monitored. The following control settings require a restart:

- **X10 and X16:** Baud Rate and Mode
- **Network:** IP Address, Gateway Address and Subnet
- **IndraLogic:** Handshaking

After changes are made to the above settings and the OK button is clicked, VisualMotion checks the current phase of the system and displays the following message when in phase 4.

Note: The user must manually reset power to the control or switch to and from parameter mode for the changes to take effect.

If the system is in parameter mode (phase 2), VisualMotion can perform a software restart of the control.



ControlReset.tif

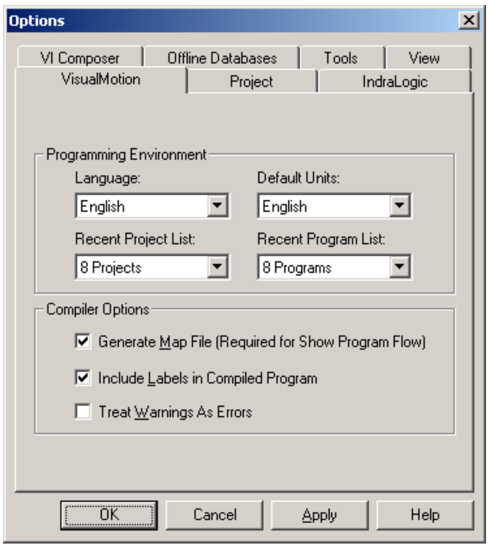
Fig. 13-71: Control Resetting

Options

The ***Options*** menu selection is used to set VisualMotion Toolkit programming environment settings, default project settings, data to be shared with IndraLogic and HMIs as well as the configuring of executable files.

VisualMotion Options

The *VisualMotion* tab is available in service or project mode and provides options for the VisualMotion programming environment and compiler:



VisualMotion_options.tif

Fig. 13-72: VisualMotion Options

Language Choose between English and Deutsch (German). This is the language used for displaying menus and tools in VisualMotion 11 **only**. Language changes do not take effect for VisualMotion windows that are currently opened. Any opened tool window must be closed and reopened for language settings to take effect.

Default Units Choose between English and Metric to set the units of measurement that will be used as the default when placing icons that have units selection. *For example:* Single Axis Setup

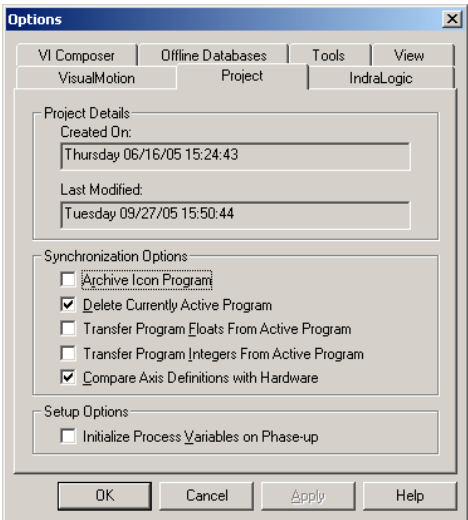
Note: This selection does not modify any existing icons in a program but is used as the default unit when new icons are added.

Recent Project / Recent Program List: The **Recent Project** and **Recent Program List** drop-down lists set the maximum number of projects and programs displayed when opening a new instance of VisualMotion Toolkit. Refer to *Open Existing Project* on page 13-4 for details.

Compiler Options: By default the **Generate Map File** and Include **Labels in Compiled Program** are set. Map files are used to show the program flow of a running project in online mode. Warnings can also be treated as errors during compilation to halt the compiler process and require correction.

Project Options

The *Project* tab is available in project mode and provide details for the currently opened project and allows the setting of default synchronization options:



Project_options.tif

Fig. 13-73: Project Options

Project Details: This read-only section provides details as to when the current project was created and last modified.

Synchronization Options: These options set the default settings for the *Synchronizing Project Data* window when switching to online mode.

Option	Description
Archive Icon Program	used to download the current icon source file (*.str) to the control's memory
Delete Currently Active Program	used to delete the current active program from the control's memory
Transfer Program Floats or Integers From Active Program	used to transfer program floats and Integers from the active program on the control to the program that will be downloaded
Compare Axis Definitions with Hardware	used to compare the drive hardware defined in the project with that of the control

Table 13-11: Icon Program Options

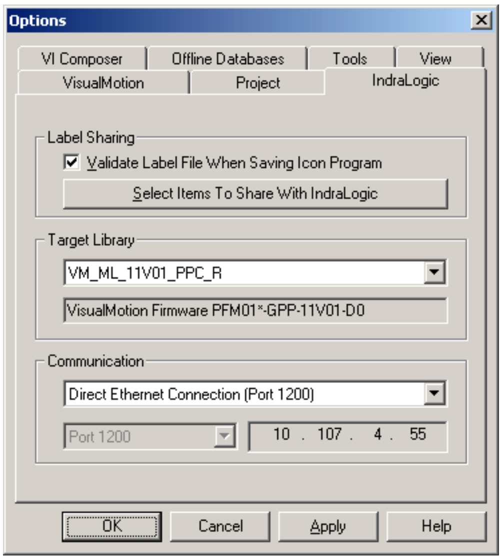
Setup Options

When set, the **Initialize Process Variables on Phase-up** option sets the project to automatically select all selectable process variables when a process is configured under ***Setup ⇒ Processes.*** Those variables that are selected will be initialized by the system. This option only affects those processes that were added after the option was set. Process variables can also be unselected to allow them to retain their values. Refer to *Defining Process Variables to Retain Values* in section 4.4, *Setup Processes*, of volume 1.

Note: Once set, this option will retain it's setting for all projects.

IndraLogic Options

The *IndraLogic* tab is available in project mode and is used to identify VisualMotion program labels that will be shared with IndraLogic, the target library that will be available based on control firmware and the communication that is used for IndraLogic.



IndraLogic_options.tif

Fig. 13-74: IndraLogic Options

Label Sharing

Selecting the **Select Items to Share with IndraLogic** button allows the user to select only those labels that will be used with IndraLogic when IndraLogic is launched from within VisualMotion Toolkit. Refer to chapter 7, *VisualMotion Variables*, in volume 1 for details.

Target Library

This option allows the user to upload functionality specific IndraLogic libraries based on the selected target firmware. When creating a new project, select the firmware that match the firmware on the control. Earlier firmware versions can be selected when programming for control's that are using an older firmware version. In any case, match the selection to the target control firmware.

Communication

The communication option is used to select a communication channel that will be used by IndraLogic for communicating with the control when launched from VisualMotion Toolkit. The user can select from the following types:

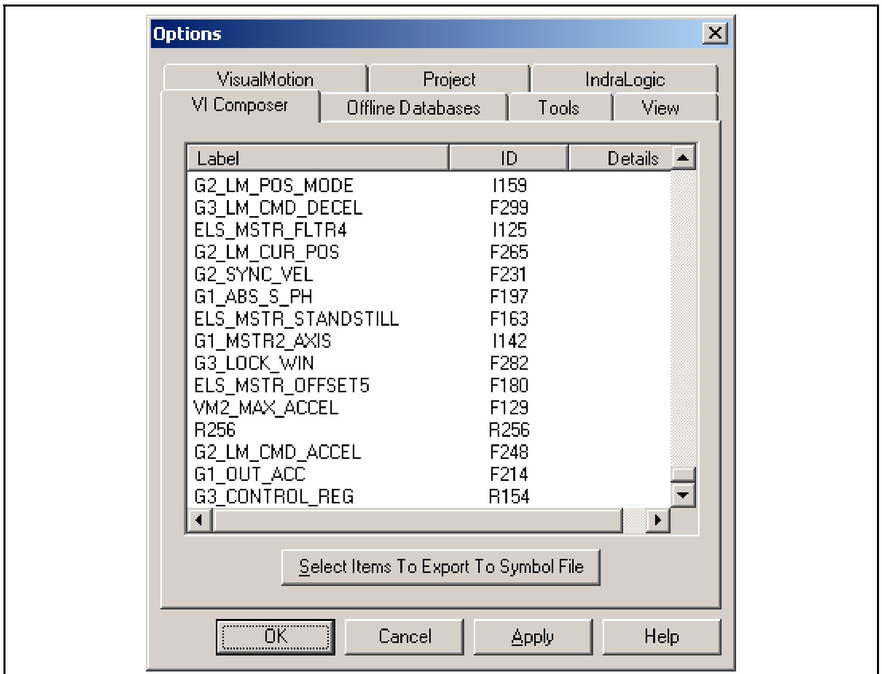
Type	Description
VisualMotion Connection (SCP)	IndraLogic's communication parameter will be set to SCP VM
Direct RS232 Connection (X16)	When online, the X16 port is automatically set to *IndraLogic*. In offline mode, the parameter can be downloaded when switching to online mode. The user must select the COM port connected to the control.
Direct EtherNet Connection (Port 1200)	When online, the control's IP address is displayed. In offline mode, the IP address saved with the project is displayed.
User Configured	This option allows the user to use the communication channel(s) that may already exists in IndraLogic.

Table 13-12: IndraLogic Communication Channels

For a complete description of the IndraLogic communication channels, refer to chapter 7, *IndraLogic Communication Channels*, in volume 1.

VI Composer Options

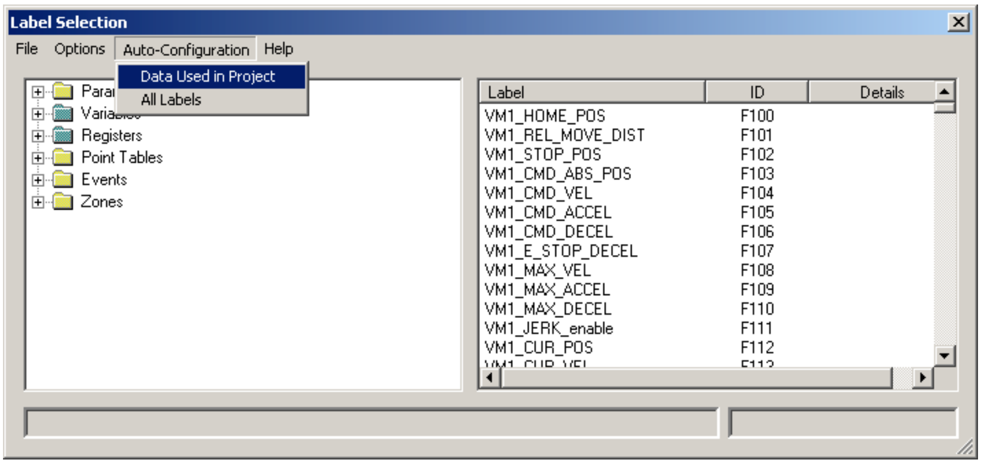
The *VI Composer* tab is available in project mode and allows the user to select VisualMotion data types that will be used to create a symbol file for use by the VI Composer software. The VI Composer software uses the symbol file and configures the access to the VisualMotion data types for an HMI device. Initially, the VI Composer window is blank.



VI_Composer_options.tif

Fig. 13-75: VI Composer Options

Clicking on the **Select Items To Export To Symbol File** button opens the *Label Selection* window. The functionality of the *Label Selection* window is similar to that of *Label Sharing* under the *IndraLogic* tab.

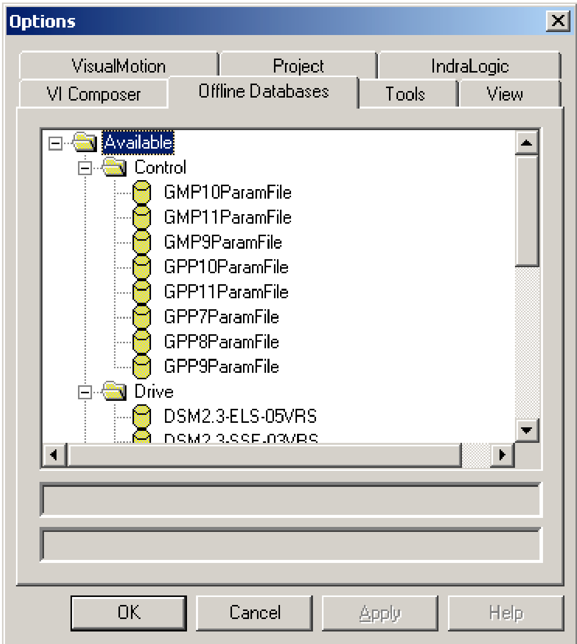


VI_Composer_label_select.tif

Fig. 13-76: Label Selection for VI Composer

Offline Database

The Offline Database tab contains a listing of all firmware databases for the control, drives, and I/O devices that are currently installed on the PC. Offline firmware databases are installed to support offline programming. These databases are accessed, for example, when setting up axes or launching the Parameter Overview tool and modifying control or drive parameters in offline mode.



DatabaseOffline.tif

Fig. 13-77: Firmware Databases in Offline Mode

Note: During the initial VisualMotion installation, all firmware databases that are supported with the current release of VisualMotion are installed.

In online or service mode, the system scans the Sercos ring for any control, drive or I/O devices that may contain newer firmware than what exists on the PC. If found, the following color scheme defines the current status of the databases:

Color	Description
Yellow	**In offline mode**, all databases installed on the PC are listed. **In online or service mode**, all databases detected on the PC but not found on the Sercos ring are listed.
Green	**In online or service mode**, the database listed on the PC was detected on the Sercos ring and is current with the connected device. Not available in offline mode.
Blue	**In online or service mode**, a difference was detected between the database on the PC and that of the specific online device. Not available in offline mode.
Red	**In online or service mode**, a database was detected on the Sercos ring that does not exits on the PC. This database is listed under a folder named Missing. Not available in offline mode.

Table 13-13: Offline Database Color Scheme

Note: The detection of firmware versions in this tool is not project specific. It simply displays the firmware databases that are installed on the PC and how current their firmware is with the devices detected on the Sercos ring.

Tool Tip Information

When placing the cursor over a green or blue firmware database, a tool tip will display the firmware versions found on both the device and on the PC. A blue firmware database may display the same firmware versions but might have some differences in parameter attributes that might require updating.

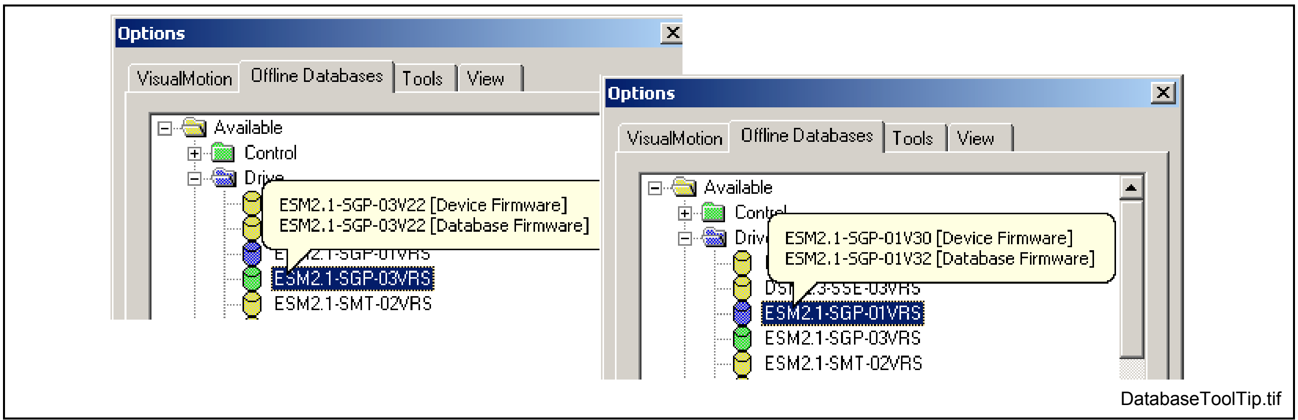


Fig. 13-78: Tool Tip Information

When placing the cursor over a red (missing) firmware database, the tool tip displays the devices that were detected on the Sercos ring containing a firmware version that was not found on the PC.

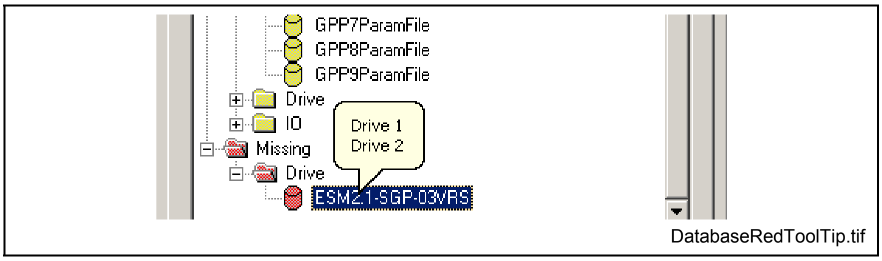


Fig. 13-79: Tool Tip Displaying Missing Firmware Database

Creating a Firmware Database

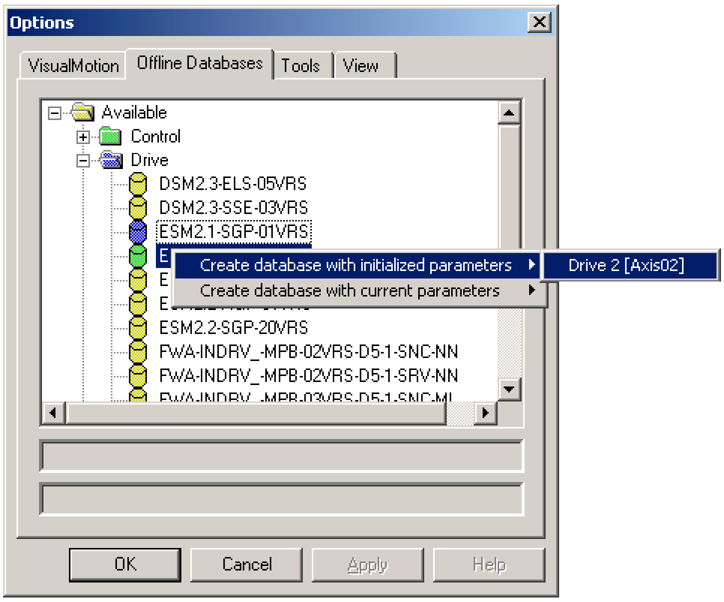
Firmware databases can be created only in online mode for a database displaying a color other than yellow. Right clicking over a green, blue, or red firmware database allows the user to create a database file on the PC from the firmware data detected on the Sercos device.

The following choices are available:

Right Click Selection	Description
Create database with initialized parameters	Creates an offline database using system default parameter values.
Create database with current parameters	Creates an offline database using the current parameter values found on the device.

Table 13-14: Creating a Firmware Database

When creating a firmware database for an existing green or blue colored database, the database on the PC will be replaced with the firmware data on the device based on the right click selection. Red, of course, means that the database does not exists on the PC, so it is simply created on the PC.

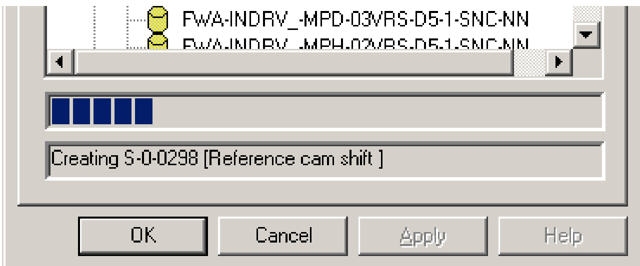


DatabaseCreate.tif

Fig. 13-80: Creating a Firmware Database

Note: Creating databases with initialized parameters will erase all user created or initialized data that exists on the device such as CAMs and point tables.

Once initiated, the *Offline Database* window displays the running status of the creation at the bottom of the window.



CreatingDatabase.tif

Fig. 13-81: Creating Firmware Database

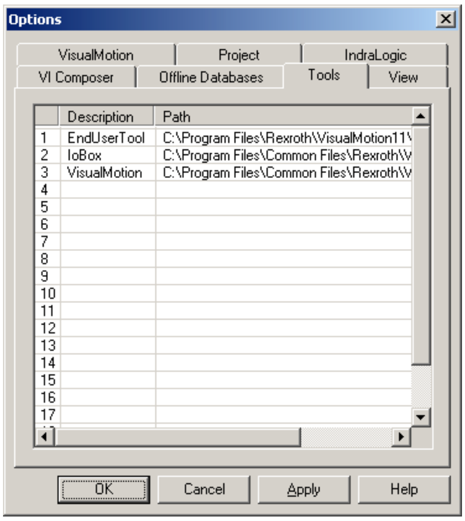
Rexroth Bosch Group

Note:	Database creation for Sercos devices such as IndraDrive can take a long time to complete. This is due to the many parameter sets that exists in IndraDrive controllers.

Tools Options

The *Tools* tab is available in service or project mode and allows the user to add any executable file (*.exe) for quick launching from the ***Tools ⇒ Register Tools*** menu selection. A maximum of 20 programs can be listed. To add a executable file to the list:

1. Double-click or right-click on an available *Path* field.
2. Locate the executable file and select it.
3. Double-click on the *Description* field and enter a name.

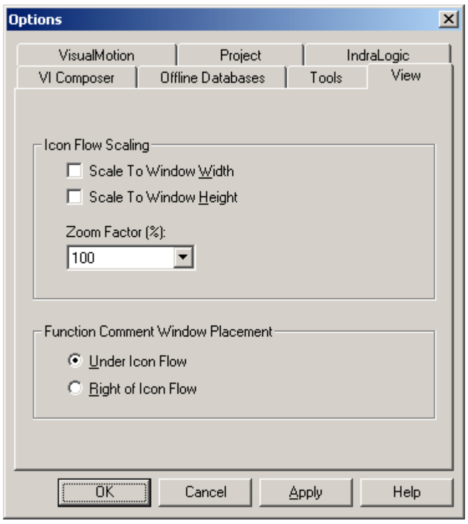


Tools_options.tif

Fig. 13-82: Tools Option for Registered Tools

View Options

The *View* tab is available in service or project mode and allows the user to change to appearance of the VisualMotion Toolkit programming environment.



View_options.tif

Fig. 13-83: View Options for VisualMotion Toolkit

Icon Flow Scaling

These options are used to scale the size of the icons used in the program. The user can size the icons to fit the width of the window, the height of the window or to a specific percentage from 25 to 200.

Function Comment Window Placement

The comment field can be selected to appear under the icons (default) or to the right of the icon program workspace.

13.11 The Window Menu

The ***Window*** menu selection is used to manage the window display of VisualMotion tools, such as I/O Setup, Variables, etc., while opened. The submenu items within the Window menu are grayed-out when no VisualMotion Tool is opened.

Note:	The Window menu is only available in project mode.

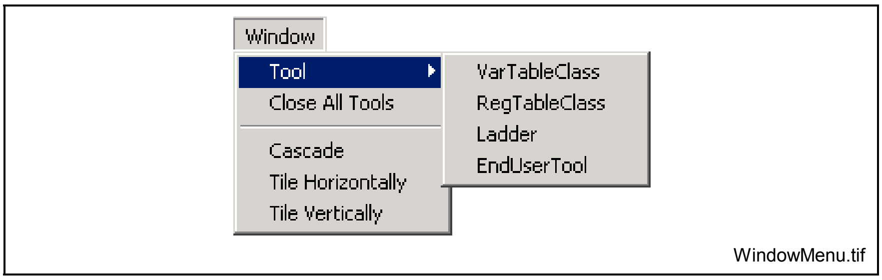


Fig. 13-84: Window Menu

Tool

This menu selection is used to quickly Activate any currently opened VisualMotion Runtime Tool.

Close All Tools

This menu selection closes all VisualMotion Tools that are opened.

Cascade

This menu selection cascades all opened VisualMotion Tools aligning the upper left-hand corner of the title bar.

Tile Horizontally

This menu selection tiles all opened VisualMotion Tools horizontally.

Tile Vertically

This menu selection tiles all opened VisualMotion Tools vertically.

13.12 The Help Menu

The **Help** menu provides assists to users in the form of an online help system. Any drive help system registered on the PC will be displayed as Registered Help.

Getting Started

Selecting **Help** ⇒ **Getting Started** or pressing **<F1>** opens the VisualMotion Help system.

Search

Selecting **Help** ⇒ **Search** opens the help's index search window into which you can type a keyword to go directly to a specific help topic.

Registered Help

During the installation of VisualMotion Toolkit, any Rexroth help system (VisualMotion or Drive) found registered in the PC's registry is added to the list. Any new help system that is installed will be displayed the next time VisualMotion Toolkit is opened.

Sercos Drive Parameters

The Parameter Overview window uses this information to display context-sensitive help for a specific drive parameter. From the Parameter Overview window, the user can double-click on a specific parameter to open the Drive Parameter Edit window. Pressing F1 launches the specific help topic for the selected drive parameter. If you do not have the correct help files for your drive, they can be requested from a Bosch Rexroth office.

About VisualMotion

Selecting ***Help*** ⇒ ***About VisualMotion*** displays VisualMotion Toolkit version, licensed and Contact information. For a listing of Bosch Rexroth service and Support locations throughout the World, click on the **Support** button.

14 Icon Descriptions

14.1 Overview

VisualMotion icons are grouped into five (5) icon palettes. Icons are selected and placed in the workspace and connected to create a logical program flow. This chapter describes how to setup icons and other VisualMotion commands.

Working with VisualMotion Toolkit Icon Palettes

VisualMotion Toolkit icon palettes are displayed below the *Project Navigator* window. Five standard palettes are provided for Initialization, Single, Coordinated, ELS and Utility icons. Icon palettes can be selected from the ***View*** ⇒ ***Icon Palette*** menu selection or by clicking on an icon tab, just below the icons in the palette. The initialization icon palette is available only when the **Initialization Task** is selected from the Project Navigator. Icons are selected from the palette using a single click of the left mouse button. The selected icon is placed in the VisualMotion workspace by positioning the cross-hair cursor over the grid area where you want the icon to appear and clicking once.

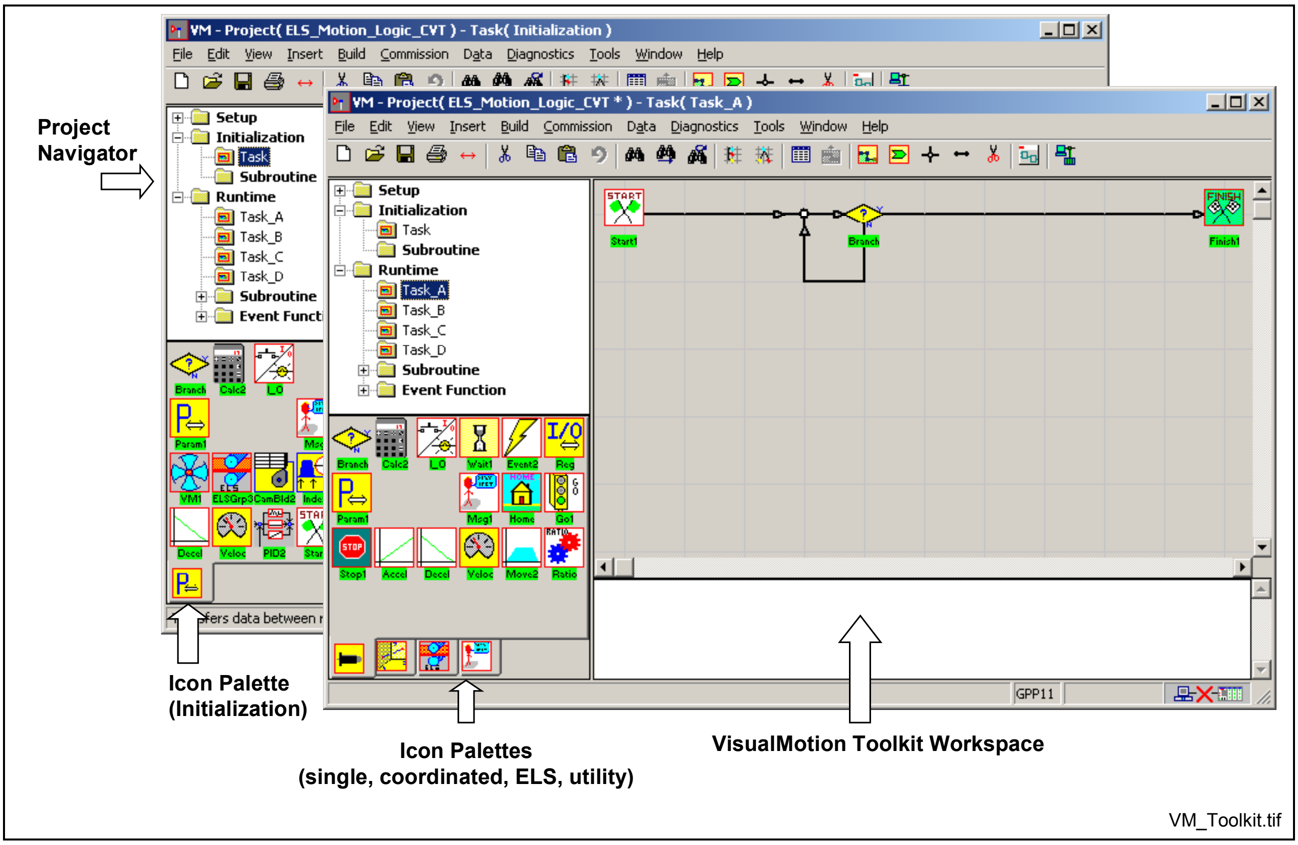


Fig. 14-1: Selecting Icon Palettes

A set of frequently used toolbar buttons is always visible above the workspace regardless of the icon palette selected.

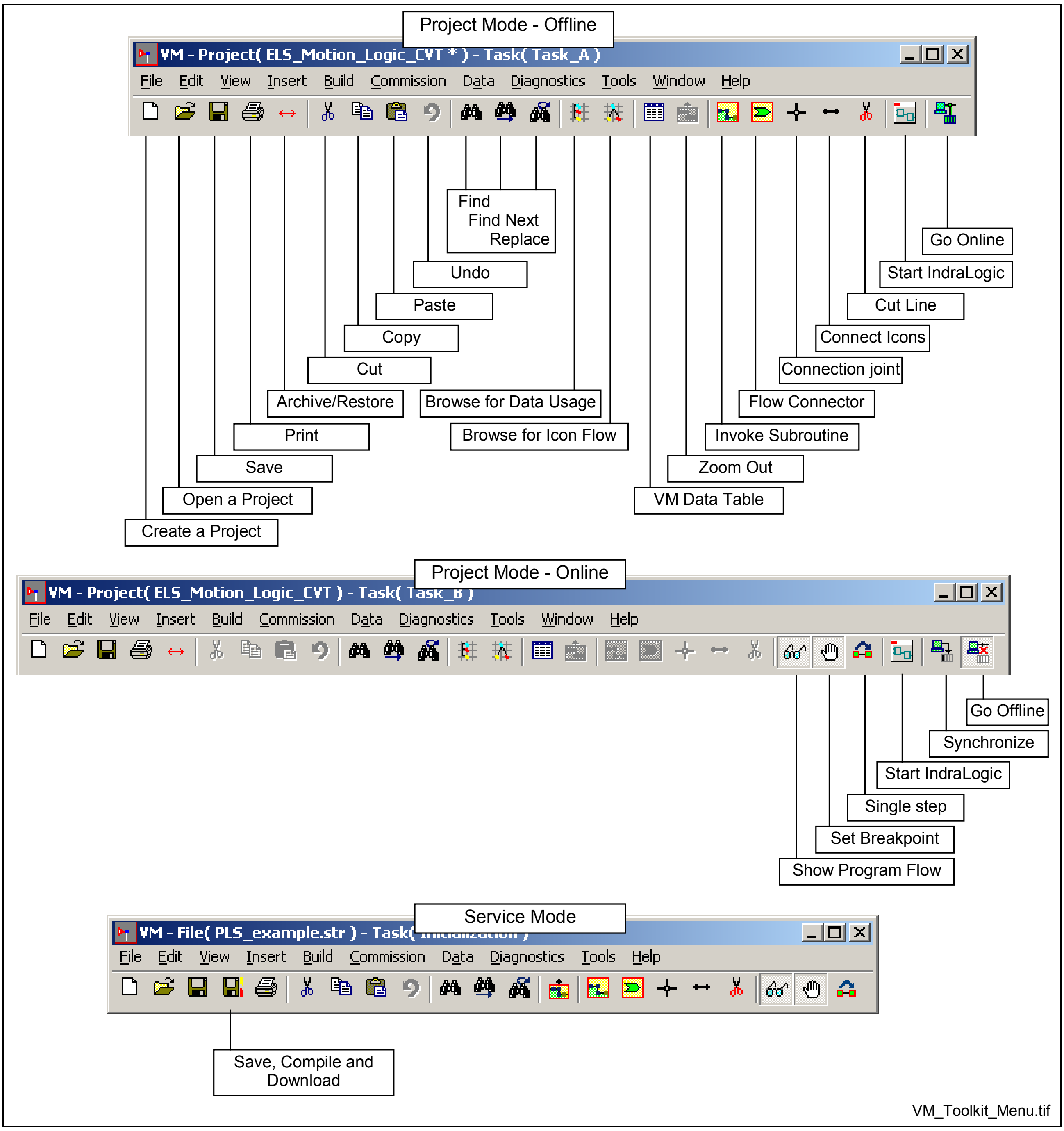
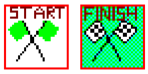


Fig. 14-2: VisualMotion Toolkit Program Menu and Icon Button Bar

Start1 and Finish1 Icons



Each VisualMotion task icon program must begin and end with a Start1 and Finish1 icon. These icons are located in the Utility Icon Palette. When a new project is created, the Initialization task is displayed with a Start1 and Finish1 icon. VisualMotion Toolkit 11 can automatically place a Start1 and Finish1 icon when an unused Task (A-D) is selected from the ***View*** ⇒ ***Task*** menu selection or Project Navigator. The following figure is displayed when selecting a task that does not contain icons.

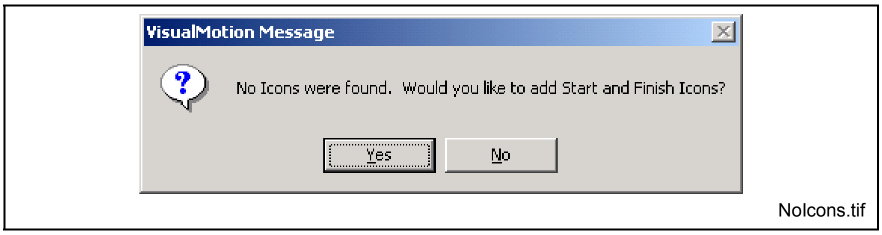


Fig. 14-3: No Icons Found

Selecting Icons

Single icons are selected by clicking and releasing the left mouse button over the icon. This process creates a red box around the icon, indicating that the icon has been selected.

A group of icons is selected in the same manner. While clicking and holding the left mouse button, drag the cursor and create a window around the icons. Any complete line connections that are included within the selection window will also be selected.

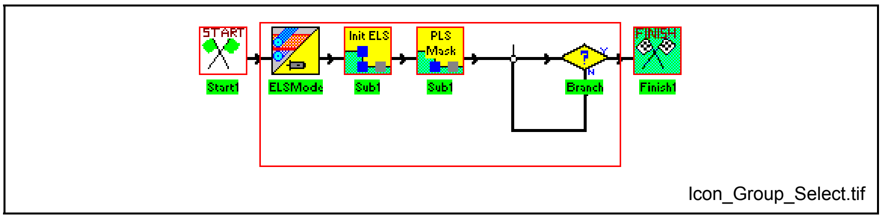


Fig. 14-4: Selecting Program Segment

Cut, Paste and Copy an Icon

Once a selection has been made, the cut and paste menu selections under *Edit* can be used. Once a selection has been cut or copied, it is saved to the Windows clipboard. If the paste button is pressed, the selection will appear in the window and follow the cursor until the left mouse button is clicked.

Deleting an Icon or Selection

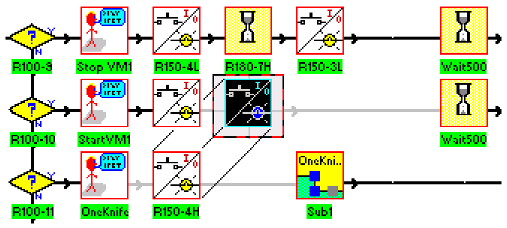
To delete or clear a selection or icon, press the delete key or select delete from the *Edit* menu after the icon(s) are selected.

Note: Deletions can be undone by selecting the Undo icon. Only one undo can be performed.

Moving an Icon or Selection

To move an icon(s), click and release the left mouse button over the icon(s), then click and hold the icon(s) again. This process places a red box around the icons and changes the appearance of the icon(s). While holding the left mouse button, drag the selection to a different position. To place the icon(s), release the mouse button.

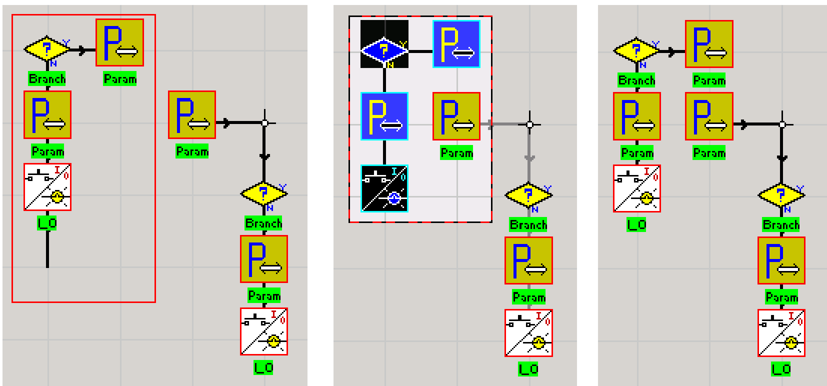
Program flow connection lines that are touching the icon(s), but are not included within the selection area, will be deleted and must be recreated once the move is complete.



Move_Icon.tif

Fig. 14-5: Moving an Icon

Moving a selection window over existing icons will not delete the icons or connection lines unless they are directly being replaced by icons or lines within the selection window. Refer to the following figure.

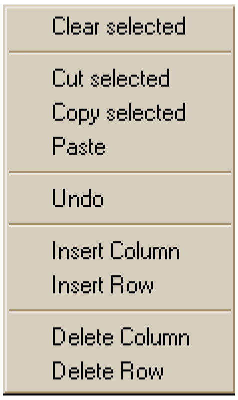


move_over.tif

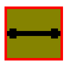
Fig. 14-6: Moving over Existing Icons

Right Mouse Click

The right mouse button is used for additional cut and paste operations. The **Clear**, **Cut** and **Copy** selected options operate the same as the described above. The **Undo** command will only undo the last operation performed.



The **Insert Column** and **Row** commands allow you to move several icons at once in order to add a new section to the program. In order to insert a column or a row, an icon or a blank section in the workspace must be selected first. The **Insert Column** command will move all the icons above, below and to the right of the selection, over one space to the right. The **Insert Row** command will move all the icons to the left, right and below the selection, down one space. The **Delete Column** and **Row** commands will delete all the icons above and below or to the left and right of the selection.

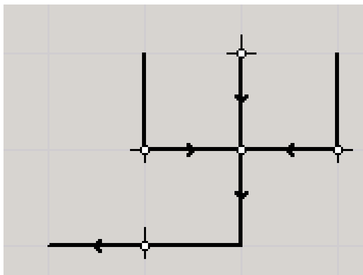
Connecting Icons 

After you have placed a number of icons on the workspace, they must be connected to indicate the program flow. Most icons have a maximum of three possible inputs and one output. The exception is the Branch icon, which has two outputs.

To draw a line, select the Line icon from the icon toolbar. Position the cursor on the first icon that you wish to connect and click. A rectangle appears, surrounding the icon. Move the cursor to the destination icon where the line is to end and click again. VisualMotion automatically draws a line from the first to the second icon, using square corners where appropriate. Arrows on the line indicate the direction of program execution. You may continue this process by clicking successive icons, without re-selecting the previous icon or the Line icon.

You may wish to manually route an interconnect to provide room for additional icons later. Under some circumstances, the Line icon's auto routing may fail to route an interconnecting line, displaying the *Connection could not be made, try connecting adjacent blocks!* window. Lines may be drawn manually by sequentially clicking on adjacent squares on the invisible workspace grid. A manually placed line may not cross another line, attempting to do so displays an error box.

A ***Join*** icon makes it possible to connect one line to another from different directions.

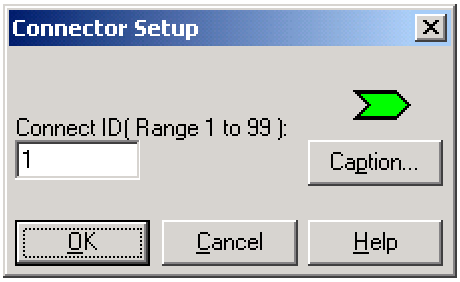


Join_Icon.tif

Fig. 14-7: Using Join Icons

A line connecting two icons may be removed by using the Scissors icon from the icon toolbar. Simply select the scissors icon, position the scissors over the line to be removed, and press the left mouse button.

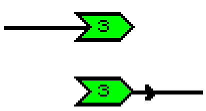
The **Add a connector** icon allows the path of a program to flow between two points that are not connected by a line. This allows complex programs to logistically fit in a relatively small workspace.



Connector_Setup.tif

Fig. 14-8: Connector Icon Setup

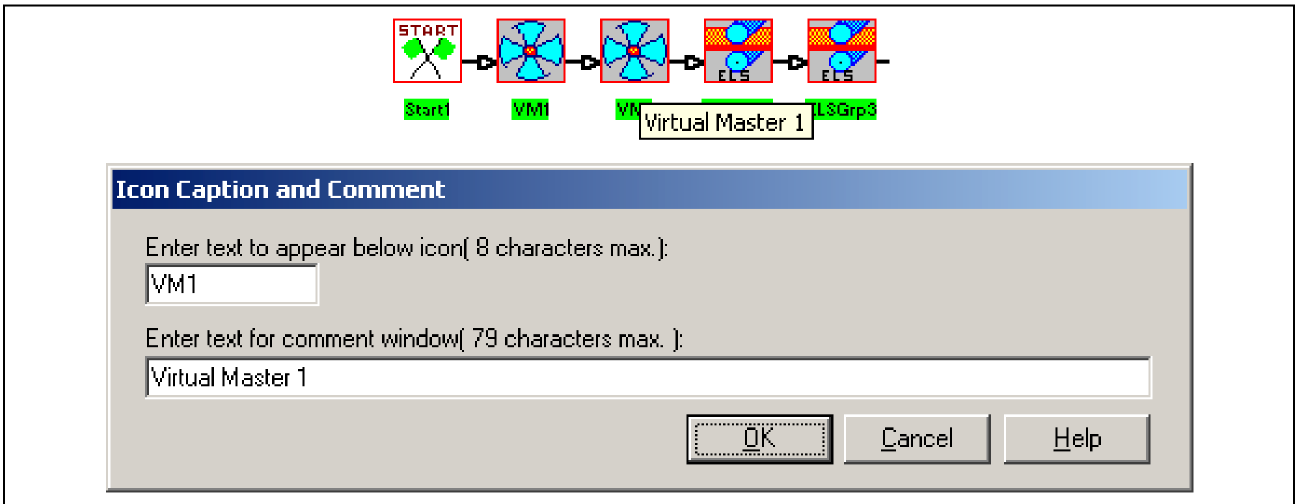
When the icon is selected and placed in the workspace a *Connector Setup* window will open and allow you to assign the icon with a Connect ID number (from 1 to 99).



ınector icon with the same Connect ID number can be placed anywhere within the same task. The order of program execution will jump from the first connector icon to the second one.

Icon Labels and Comments

Each Icon has a label which can be displayed when "Icon Labels" is selected from the ***View Menu***. Descriptive comments, up to 80 characters long, are automatically added by VisualMotion describing the selects made in the icon setup window. The use may also modify the default comments. When "Icon Comments" is selected, the comments will appears in a tool tip window as the cursor passes over the icon.



Icon_Caption.tif

Fig. 14-9: Icon Captions and Comments

The comments can be entered by pressing the ***Caption…*** button within the icon window. They are used to document the purpose of the icon relative to the program. The Caption under each icon can also be changed independently.

14.2 VisualMotion Toolkit Icons

VisualMotion Toolkit 11 provides five (5) standard palettes for Initialization, Single, Coordinated, ELS and Utility icons.

Initialization Icon Palette

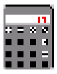
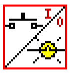
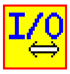
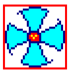
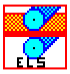
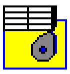
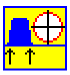
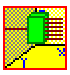
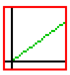
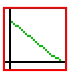
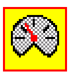
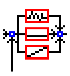
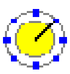
Icon	Name	Function	Valid Tasks Init.	Valid Tasks A-D	Page
	Branch	Directs program flow	x	x	14-13
	Calc3	Calculation and initialization	x	x	14-15
	I_O	Sets an I/O register bit	x	x	14-43
	Reg Register Transfer	Transfer data between registers and variables	x	x	14-73
	Param1 Parameter Transfer	Transfers a control or drive parameter, sets bit(s) in control or drive parameter, and Initiates drive resident commands	x	x	14-48
	Msg1 Message	Sets status or diagnostic message	x	x	14-47
	SlipMon Slip Monitoring	Monitors two master signals within a specific window	x		14-74
	ELSMstr2 ELS Master Runtime	Assigns ELS Masters	x		14-39
	VM1 Virtual Master Runtime	Assign Virtual Masters	x		14-84
	ELSGrp3 ELS Group Runtime	Assigns axes in an ELS Group	x		14-37
	CamBld2 CAM Build	Builds a CAM table	x	x	14-28
	Index1	Initializes a CAM indexer	x	x	14-32
	CoordArt2 Coordinated Articulation	Sets and initializes coordinated articulation	x		14-34
	Accel Acceleration	Sets the acceleration of an axis	x	x	14-12
	Decel	Sets the deceleration of an axis	x	x	14-35
	Veloc Velocity	Sets velocity for non-coordinated motion	x	x	14-83
	PID2	Installs and initializes a PID loop	x		14-69
	PLS2	Sets up a control based PLS	x	x	14-71

Table 14-1: Initialization Icon Palette

Rexroth
Bosch Group

Single Axis Icon Palette

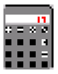
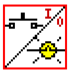
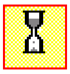
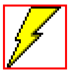
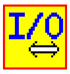
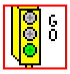
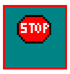
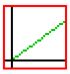
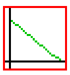
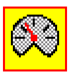
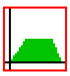
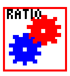
Icon	Name	Function	Valid Tasks Init.	Valid Tasks A-D	Executed	Page
	Branch	Directs program flow	x	x	Runtime	14-13
	Calc3	Calculation and initialization	x	x	Runtime	14-15
	I_O	Sets an I/O register bit	x	x	Runtime	14-43
	Wait1	Conditionally holds program execution		x	Runtime	14-85
	Event2	Event Setup Box		x	Runtime	14-41
	Reg Register Transfer	Transfer data between registers and variables	x	x	Runtime	14-73
	Param1 Parameter Transfer	Transfers a control or drive parameter, sets bit(s) in control or drive parameter, and Initiates drive resident commands	x	x	Runtime	14-48
	Msg1 Message	Sets status or diagnostic message	x	x	Runtime	14-47
	Home	Initiates drive homing procedure		x	Runtime	14-43
	Go1	Enables an axis (axes)		x	Runtime	14-42
	Stop1	Halts motion		x	Runtime	14-79
	Accel Acceleration	Sets the acceleration of an axis	x	x	Runtime	14-12
	Decel	Sets the deceleration of an axis	x	x	Runtime	14-35
	Veloc Velocity	Sets velocity for non-coordinated motion	x	x	Runtime	14-83
	Move2	Axis position move		x	Runtime	14-45
	Ratio	Sets the ratio between two axes		x	Runtime	14-71

Table 14-2: Single Icon Palette

Coordinated Motion Icon Palette

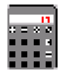
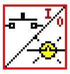
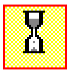
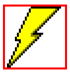
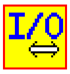
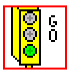
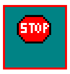
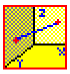
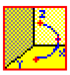
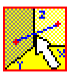
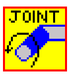
Icon	Name	Function	Valid Tasks Init.	Valid Tasks A-D	Executed	Page
	Branch	Directs program flow	x	x	Runtime	14-13
	Calc3	Calculation and initialization	x	x	Runtime	14-15
	I_O	Sets an I/O register bit	x	x	Runtime	14-43
	Wait1	Conditionally holds program execution		x	Runtime	14-85
	Event2	Event Setup Box		x	Runtime	14-41
	Reg Register Transfer	Transfer data between registers and variables	x	x	Runtime	14-73
	Param1 Parameter Transfer	Transfers a control or drive parameter, sets bit(s) in control or drive parameter, and Initiates drive resident commands	x	x	Runtime	14-48
	Msg1 Message	Sets status or diagnostic message	x	x	Runtime	14-47
	Home	Initiates drive homing procedure		x	Runtime	14-43
	Go1	Enables an axis (axes)		x	Runtime	14-42
	Stop1	Halts motion		x	Runtime	14-79
	Path	Coordinated straight line move		x	Runtime	14-68
	Circle	Coordinated circular move		x	Runtime	14-33
	Position	Gets position of coordinated move		x	Runtime	14-70
	Joint	Six axis coordinated move		x	Runtime	14-44

Table 14-3: Coordinated Motion Icon Palette

ELS Icon Palette

Icon	Name	Function	Valid Tasks		Executed	Page
			Init.	A-D		
	Branch	Directs program flow	x	x	Runtime	14-13
	Calc3	Calculation and initialization	x	x	Runtime	14-15
	I_O	Sets an I/O register bit	x	x	Runtime	14-43
	Wait1	Conditionally holds program execution		x	Runtime	14-85
	Event2	Event Setup Box		x	Runtime	14-41
	Reg Register Transfer	Transfer data between registers and variables	x	x	Runtime	14-73
	Param1 Parameter Transfer	Transfers a control or drive parameter, sets bit(s) in control or drive parameter, and Initiates drive resident commands	x	x	Runtime	14-48
	Msg1 Message	Sets status or diagnostic message	x	x	Runtime	14-47
	Home	Initiates drive homing procedure		x	Runtime	14-43
	ELSMode Mode Change	Switches phase or velocity of ELS axis		x	Runtime	14-40
	ELSAdj1 ELS Adjust	Shifts phase or velocity of ELS axis		x	Runtime	14-35
	Cam	Assigns a CAM to an axis		x	Runtime	14-26
	CamBld2 Cam Build	Builds a CAM table	x	x	Runtime	14-28
	CamAdj Cam Adjust	Phase shift a CAM axis		x	Runtime	14-27

Table 14-4: ELS Icon Palette

Utility Icon Palette

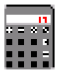
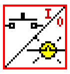
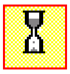
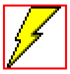
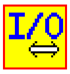
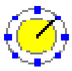
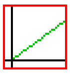
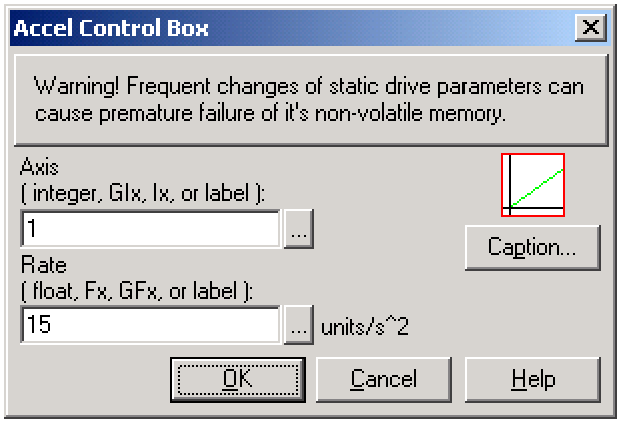
Icon	Name	Function	Valid Tasks Init.	A-D	Executed	Page
	Branch	Directs program flow	x	x	Runtime	14-13
	Calc3	Calculation and initialization	x	x	Runtime	14-15
	I_O	Sets an I/O register bit	x	x	Runtime	14-43
	Wait1	Conditionally holds program execution		x	Runtime	14-85
	Event2	Event Setup Box		x	Runtime	14-41
	Reg Register Transfer	Transfer data between registers and variables	x	x	Runtime	14-73
	Param1 Parameter Transfer	Transfers a control or drive parameter, sets bit(s) in control or drive parameter, and Initiates drive resident commands	x	x	Runtime	14-48
	Msg1 Message	Sets status or diagnostic message	x	x	Runtime	14-47
	Start1	Sets the start of program flow in each task	x	x	Runtime	14-75
	Finish1	Sets the end (finish) of the program flow in each task.	x	x	Runtime	14-42
	PLS2	Sets up a control based PLS	x	x	Runtime	14-71

Table 14-5: Utility Icon Palette

Accel



The Accel icon is used to set or change the acceleration of a single non-coordinated axis.



Accel_Icon.tif

Fig. 14-10: Accel Icon Setup

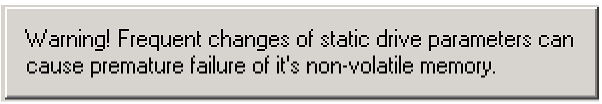
Axis indicates the axis to accelerate. The axis may be entered as an integer constant, integer variable, global integer variable or an equivalent label.

Rate specifies the acceleration rate in units per second squared. The entry may be a float constant, variable, global variable or an equivalent label. The variable must be greater than 0.

Units: Axis Definition

Single Axis Linear - inches	inches/s^2
Single Axis Linear - mm	mm/s^2
Single Axis Rotary	rads/s^2
Velocity Mode	rads/s^2

Clicking the browse button to the right of each data field opens the VM Data Table.



The warning message that frequent changes to the static drive parameters can damage the non-volatile memory, applies only to the following drives with all versions of firmware:

- Rexroth EcoDrive 01
- Rexroth Diax 04
- Rexroth Diax 03

It is possible to prevent damage to the EEPROM by changing the default setting of the drive parameter S-0-0269 (Buffer Mode) from 0 to 1. The change forces the Sercos control to disable the parameter buffer on every power up sequence, limiting the number of times parameter changes are written to memory.

Note: No memory problem occurs in the following version of Rexroth EcoDrive 03:

ECODR3-SGP-01, ECODR3-SMT-01, ECODR3-SMT-02

ECODR3-SGP-03, ECODR3-SGP-20, ECODR3-SMT-20

Branch

A Branch icon re-directs the program flow depending upon a true/false logical value.

Four radio buttons at the bottom of the window permit selection of one of four Branch icon graphics, each with a different positioning of the branch test outputs.

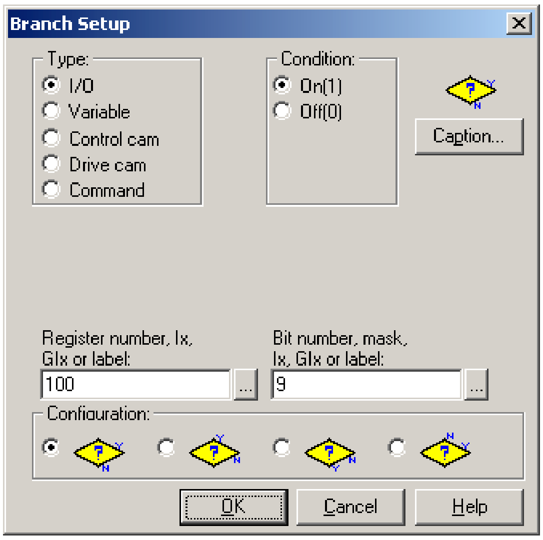
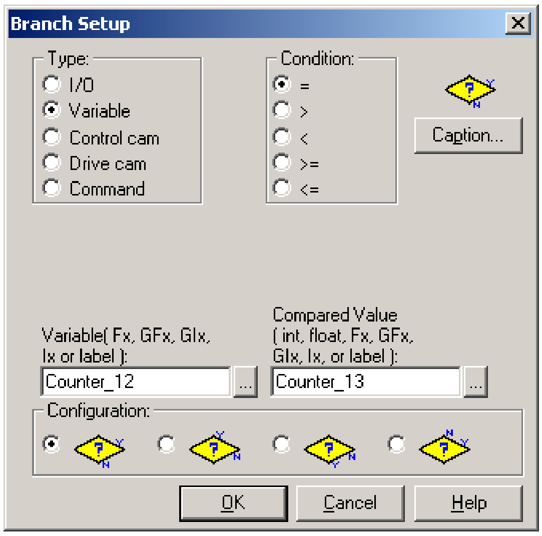


Fig. 14-11: Branch Icon Setup for I/O

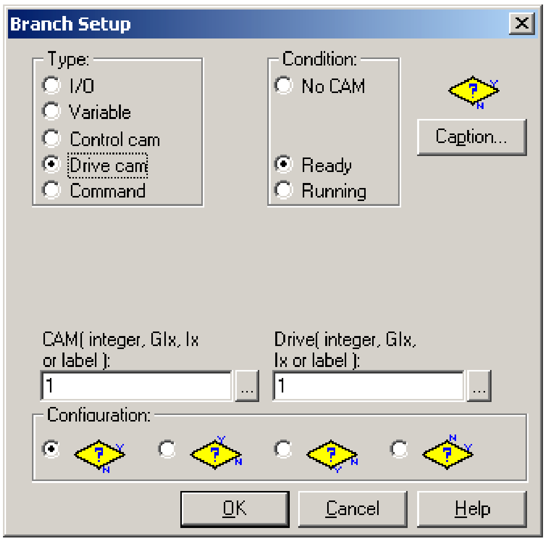
Selecting an I/O type branch permits testing a bit in a specified I/O register for a logical (true/false) condition. The I/O register for comparison is specified by an integer or an equivalent label. A bit mask only permits testing of selected bits within the specified register to be tested. The bit mask may be entered as an integer or equivalent label. I/O tests are true/false only the equality or inequality relationships are not allowed.



Branch_Setup_Var.tif

Fig. 14-12: Branch Setup Icon for Variable

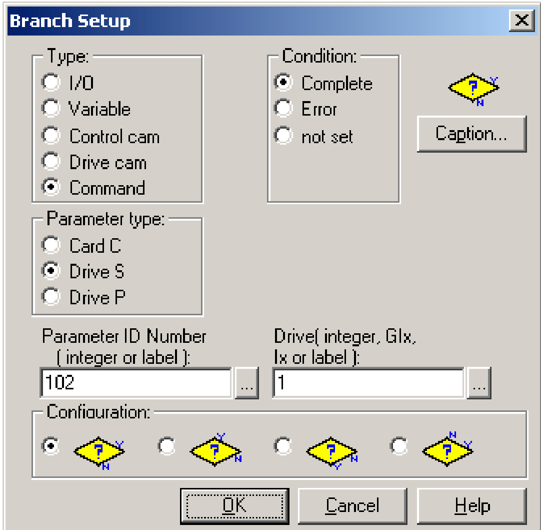
Selecting Variable permits a comparison of the contents of an integer variable, float variable or an equivalent label with a compared value (integer or float constant, or an equivalent label). Clicking the browse button to the right of any field opens the VM Data Table.



Branch_Setup_Drive.tif

Fig. 14-13: Branch Icon Drive Cam Setup

Selecting Control or Drive CAM permits a specified CAM check for one of three conditions. The branch will return a yes or no value if there is No CAM or if the CAM is Ready or Running.



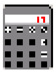
Branch_Setup_Cmnd.tif

Fig. 14-14: Branch Icon Command Setup

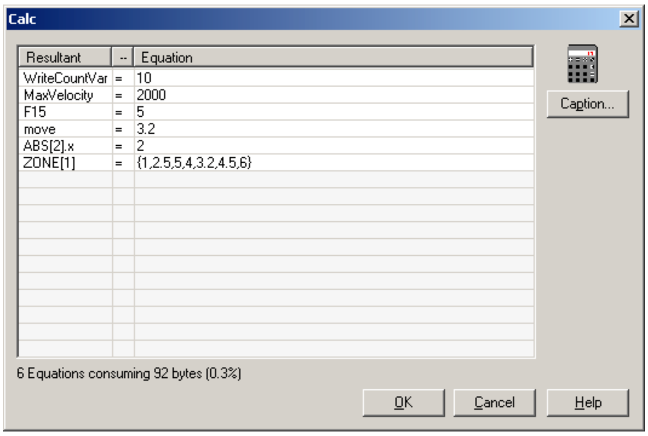
Selecting Command enables the Branch icon to monitor the condition of a Control, or Drive parameters before allowing program flow to continue.

Note: When entering a parameter ID for an S or P class drive parameter (S-0-0102), simply enter to number (102).

Calc3



The Calc3 icon can be used to create calculations within the program flow and/or to initialize variables, points, events, zones or PLS elements. By using the Calc3 icon in this fashion, project variables can be initialized with operational values, thereby eliminating the need to specify them after downloading the program to the control. The Calc3 icon supports multiple expressions within one icon declaration. This feature greatly reduces the number of Calc3 icons required in a project.



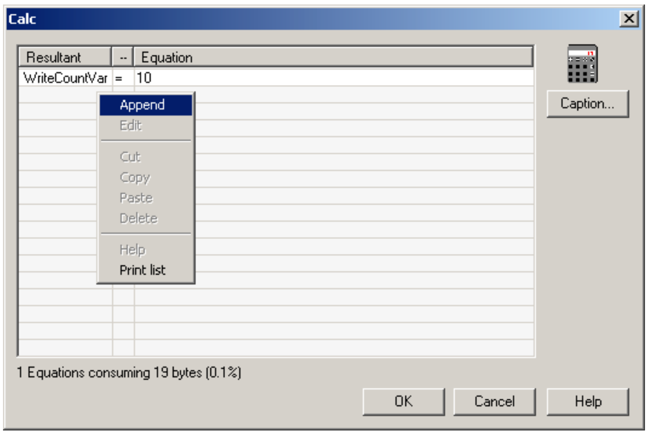
Calc3_Example.tif

Fig. 14-15: Calc3 Icon Window

Append an Expression

The number of expressions that can be appended to a Calc icon is dependent on the amount of memory consumed by the expressions. Each Calc icon is limited to 32 Kbyte. Once this limit is reached, no further expressions can be appended.

To add an expression, right click in the Calc3 window and select ***Append***.

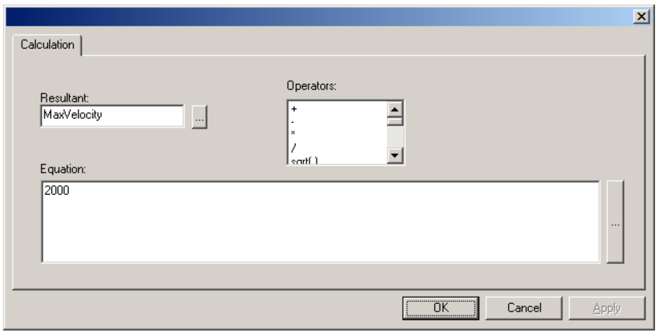


Calc3_Append.tif

Fig. 14-16: Append a Calc Expression

Calculation Expressions

The *Calculation* window is used to create the expressions that will be used in the program.



Calc3_Resultant.tif

Fig. 14-17: Calc3 Icon Expressions

The **Resultant** field is used to specify the element for a variable, point table, event, zone table or PLS element.

- Variable {`Fx,GFx,GIx,Ix,label`}
- Point table element {`ABS[x],REL[x]`}
- Event table element {`EVT[x]`}
- Zone table element {`ZONE[x]`}
- Programmable limit switch {`PLS[x]`}

Resultants can be selected or created from the VM Data Table window by clicking on the button to the right of the **Resultant** and **Equation** fields.

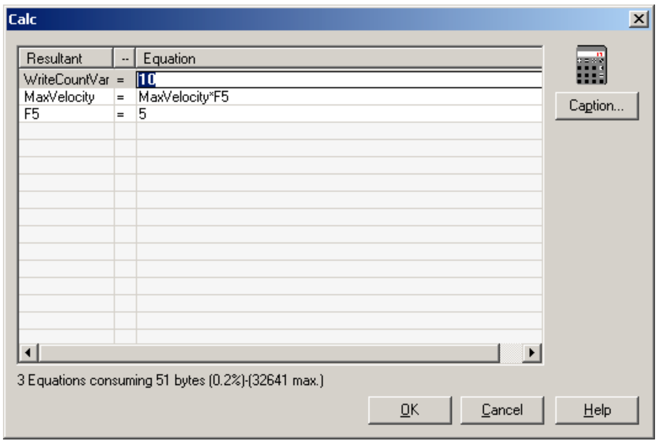
Enter the expression to be evaluated in the **Equation** field. Operators may be selected from the Operators scrolling menu or entered using the keyboard. Symbolic labels that equate to a constant may be used in equations. The equation is evaluated from left to right. Parentheses can be used to control the order of evaluation.

Note: The Operators menu includes type conversion operators for float and integer data types. These operators should be placed immediately to the left of the operand (e.g., as the equation is parsed from left to right, the compiler first obtains the value, then performs the conversion.

Although an equation can contain a combination of float variables, integer variables, and constants, the data types used for a single operation must match. Data type mismatches are especially easy to overlook when using symbolic label names that do not implicitly identify the data type. Use the "float( )" and "int( )" conversion operators to force the proper data types.

Editing Expressions

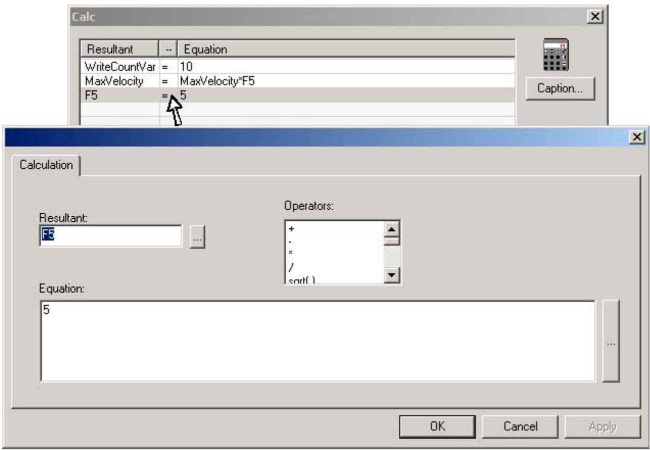
Calculation expressions can be edited inline from within the list page view or from within the *Calculation* window. To edit an expression inline, select the relevant *Resultant* or *Equation* field using the left mouse button and modify the value accordingly.



Calc3_EditInline.tif

Fig. 14-18: Editing a Calc3 Expression Inline

To edit using the *Calculation* window, double click on the equals (=) sign column of the relevant expression line, make the necessary modifications and click the **OK** button.



Calc3_EditCalc.tif

Fig. 14-19: Editing a Calc3 Expression in the Calculation Window

Note: The *Calculation* window can also be displayed with a left mouse click followed by a right mouse click over the equals (=) sign and then select Edit.

Rexroth Bosch Group

Operators (Mathematical and Logical)

The operators available for use in the Calc3 icon are listed in the following table:

<table>
<tr><td colspan="2">Arithmetic operators</td></tr>
<tr><td>+</td><td>addition (floating point or integer)</td></tr>
<tr><td>-</td><td>subtraction (floating point or integer)</td></tr>
<tr><td>*</td><td>multiplication (floating point or integer)</td></tr>
<tr><td>/</td><td>division (floating point or integer)</td></tr>
<tr><td colspan="2">Logical operators Note: Logical operations are limited to unsigned integers.</td></tr>
<tr><td>&</td><td>logical bitwise AND of two integers</td></tr>
<tr><td>|</td><td>logical bitwise Inclusive OR or two integers (101 | 011 is 111)</td></tr>
<tr><td>^</td><td>logical bitwise "exclusive or" of two integers (101 ^ 110 is 011)</td></tr>
<tr><td>not</td><td>logical bitwise inversion of an int (changes "0s" to "1s" and "1s" to "0s")</td></tr>
<tr><td><<n</td><td>Int shift "n" bits left (1-16 bits), low bits padded with 0, high bits are lost</td></tr>
<tr><td>>>n</td><td>Int shift "n" bits right (1-16 bits), high bits padded with 0, low bits are lost</td></tr>
<tr><td colspan="2">Transcendental functions Note: All transcendental functions are in radians.</td></tr>
<tr><td>sin(n)</td><td>returns the floating point sine of an integer or float</td></tr>
<tr><td>cos(n)</td><td>returns the floating point cosine of an integer or float</td></tr>
<tr><td>tan(n)</td><td>returns the floating point tangent of an integer or float</td></tr>
<tr><td>asin(n)</td><td>returns the floating point arcsine of an integer or float</td></tr>
<tr><td>acos(n)</td><td>returns the floating point arccosine of an integer or float</td></tr>
<tr><td>atan(n)</td><td>returns the floating point arctangent of an integer or float</td></tr>
<tr><td>ln(n)</td><td>returns the floating point natural logarithm (base e) of an integer or float</td></tr>
<tr><td>log(n)</td><td>returns the floating point logarithm (base 10) of an integer or float</td></tr>
<tr><td colspan="2">Exponential functions:</td></tr>
<tr><td>n**p</td><td>returns the floating point value of "n" raised to the "p" power</td></tr>
<tr><td>sqrt(n)</td><td>returns the floating point square root of an integer or float</td></tr>
<tr><td colspan="2">Conversion functions:</td></tr>
<tr><td>%</td><td>modulus (the remainder or fractional portion) of the result of the division of 2 integers or floating point numbers</td></tr>
<tr><td>int(n)</td><td>returns the integer portion of a floating point value as an integer</td></tr>
<tr><td>float(n)</td><td>returns a floating point value equal to an integer</td></tr>
<tr><td>frac(n)</td><td>returns a floating point value equal to an integer</td></tr>
<tr><td>absolute(n)</td><td>converts a positive or negative integer or float to a positive integer</td></tr>
<tr><td>bintoBCD(n)</td><td>converts a binary value to a packed BCD integer</td></tr>
<tr><td>BCDtobin(n)</td><td>converts a packed BCD to a binary value</td></tr>
</table>

Table 14-6: Mathematical operators

Integers and Floats

Integers are signed or unsigned whole numbers, such as 5 or –3. Integers cannot have a fractional component.

Floats are signed and unsigned numbers using a decimal point, such as 5.0 or -3.0.

Constants, Variables and Global Variables

Constants may be any valid integer or float, such as 3.14159. Constants cannot be modified once a user program is compiled and downloaded to the control.

Variables are names (or labels) used as a symbolic representation for an integer (`Ix`) or float (`Fx`) value. Variable can be modified after a user program is compiled and downloaded to the control by selecting ***Data ⇒ Variables***.

Global Variables are also names used to represent values for a global integer (`GIx`) or global float (`GFx`). Global variables reside in the control's RAM memory and are not retained during power-off. Global variables in user programs are not automatically initialized by VisualMotion's compiler. It is the programmer's responsibility to explicitly initialize global variables in a program.

Direct and Indirect Addressing

VisualMotion stores program variable values and data in separate areas of memory called tables, also referred to as arrays.

A variable table is addressed by using an "`I`" or "`F`" prefix before the integer number (e.g., `I5` or `F20`).

An absolute or relative points table or event table is addressed by using an "ABS", "REL" or "EVT" prefix before the table number in brackets (e.g., ABS[1]).

Direct Addressing is performed when a variable or table number is used to access the value directly. Variable labels can be used to access float and integer tables.

For Example:

`F15 = 5`	accesses the 15th entry (row) of the float table and writes a value of 5
`move = 3.2`	accesses the entry associated with the label "move" in the float table and writes a value of 3.2
`ABS[2].x = 2`	accesses the 2nd point of the absolute points table and writes a value of 2 for it's x element

Indirect Addressing is performed when a variable or table entry is access indirectly by enclosing an integer variable within square brackets. The indirect element must be an integer variable; Global or Program (`GIx,Ix,label`).

For Example:

Using an Integer Variable Indirectly	
`F[I5] = 5.0` ⇩ `F[2] = 5.0`	First, the value in `I5` is used as the entry number for the float variable If `I5=2`, then the 2nd entry (row) of the float table is accessed and a value of 5.0 is written
`ABS[I2].x = 4` ⇩ `ABS[3].x = 4`	First, the value in `I2` is used as the point in the absolute table If `I2=3`, then the 3rd point of the absolute points table is accessed and 4 is written to it's x element
Using a Label Assigned to an Integer Variable Indirectly	
`F[move] = 3.2` ⇩ `F[2] = 3.2`	First, the integer's value assigned to the label "move" is used as the entry number for the float variable If `move=I5=2`, then the 2nd entry (row) of the float table is accessed and a value of 3.2 is written
`ABS[set].x = 4` ⇩ `ABS[3].x = 4`	First, the integer's value assigned to the label "set" is used as the point in the absolute table If `set=I2=3`, then the 3rd point of the absolute points table is accessed and 4 is written to it's x element

Points Table (Coordinated Motion Variables)

Absolute (`ABS[x]`) and Relative (`REL[x]`) points, used to define a geometric path segment for coordinated motion, are stored in an absolute and relative point table. The elements used for an absolute or relative point table are listed as follows:

Resultant	Equation (Elements)
ABS[n]	= {**x,y,z,b,s,a,d,j,e1,e2,e3,e4,r,p,ya,el**}
REL[n]	= {**x,y,z,b,s,a,d,j,e1,e2,e3,e4,r,p,ya,el**}

Note: Elements in the data structure are separated by a comma(,) and no spaces are allowed between element values.

Element	Description
x	x Cartesian coordinate
y	y Cartesian coordinate
z	z Cartesian coordinate
b	blend radius
s	% of max. speed
a	% of max. acceleration
d	% of max. deceleration
j	% of jerk limiting
e1 – e4	event ID number
r	degree of roll (rate for kinematic #8)
p	degree of pitch
ya	degree of yaw
el	elbow state/coordinated axis mask

Examples of data structure initialization:

The Calc3 icon can be used to initialize all the elements of a data structure.

```
ABS[1]={5.1,4.2,2.3,1.0,55,66,77,88,1,2,3,4,3.0,4.0,5.0,1}
REL[1]={1.1,1.2,1.3,1.0,55,66,77,88,1,2,3,4,3.0,4.0,5.0,1}
```

Examples using data structure elements:

The Calc3 icon can be used to write a value to a single element in the data structure.

```
ABS[1].x = 5.1
REL[5].z = 1.3
REL[I1].y = 1.2   ; indirect addressing
```

Examples of data transfer between like structures

The Calc3 icon can be used to transfer data between like data structures.

```
ABS[1]= ABS[2]
REL[1]= REL[2]
```

Events

Events are used to run a specific process in applications. Events interrupt task motion until complete. Using the Calc3 icon, the complete event data structure or a single element can be initialized when the user program is started. The Event data structure is listed in the following table.

Resultant	Equation (Elements)
`EVT[n]`	= {**s,t,d,a,f,m**}

Note: Elements in the data structure are separated by a comma(,) and no spaces are allowed between element values.

Element	Description	Selections
s	Status The Status element is read-only and is used as a placeholder. Enter a 0 when initializing. The Calc icon can be used to read the status of this element. Example: F20 = EVT[1].s	0 = Inactive 1 = Queued 2 = Pending 3 = Executing 4 = Done 6 = Repeat
t	Type	0 = <undefined> 1 = Repeating Timer 2 = Time in Coordinated Path 3 = Percent in Coordinated Path 4 = Single Axis Distance 5 = Rotary 6 = Task Input Transition 7 = VME (pre-GPS7 only) 8 = VME (pre-GPS7 only) 9 = Feedback Capture (Probe) 10 = I/O Register Event 11 = PPC-R X1 Input Events
d	Direction	0 = Start of Move (positive edge for input event) 1 = End of Move (negative edge for input event)
a	Argument When Type = 1 ------------------> = 2 ------------------> = 3 ------------------> = 4 ------------------> = 5 ------------------> = 6 ------------------> = 9 ------------------> = 10 or 11 ---------->	This element value is entered based on the selected Type: time in ms time in ms from start/end of move % distance from start/end of move distance from start/end of single move degree to activate the event set to zero, no meaning set to zero, no meaning bit number of input signal
f	Function	Name of Event function
m	Message	text from 0-80 characters in quotation marks " ".

Example of data structure initialization:

The Calc3 icon can be used to initialize all the elements of a data structure.

```
EVT[1]={0,1,0,20,IO_On,"Enables IO"}
```

Examples using data structure elements:

The Calc3 icon can be used to write a value to a single element in the data structure.

```
EVT[1].t = 1

EVT[1].m = "Enables IO"

EVT[I1].f = IO_On        ; indirect addressing
```

Example of data transfer between like structures

The Calc3 icon can be used to transfer data between like data structures.

```
EVT[1]= EVT[2]
```

Zone Tables

Zones are used in Coordinated motion programs to describe a volume of space where motion of any kind is prevented. Each zone is defined by the { x, y, z } coordinates of two opposite corners (Point 1 and Point 2) of a cube in space. A Zone status is defined as active or inactive for one or more tasks (or all tasks).

Resultant	Equation (Elements)
ZONE[n]	= {**s,a,b,c,d,e,f**}

Note: Elements in the data structure are separated by a comma(,) and no spaces are allowed between element values.

Element	Description	Selections
s	Status	1 = All Task 2 = Task A 4 = Task B 8 = Task C 16 = Task D
a	x Cartesian coordinate, Point 1	
b	y Cartesian coordinate, Point 1	
c	z Cartesian coordinate, Point 1	
d	x Cartesian coordinate, Point 2	
e	y Cartesian coordinate, Point 2	
f	z Cartesian coordinate, Point 2	

Example of data structure initialization:

The Calc3 icon can be used to initialize all the elements of a data structure.

```
ZONE[1]={1,2.5,5,4,3.2,4.5,6}
```

Examples using data structure elements:

The Calc3 icon can be used to write a value to a single element in the data structure.

```
ZONE[1].a = 2.5
ZONE[I1].b = 5    ; indirect addressing
```

Example of data transfer between like structures

The Calc3 icon can be used to transfer data between like data structures.

```
ZONE[1]= ZONE[2]
```


Rexroth
Bosch Group

PLS Data

Unlike variables, points tables and event tables, PLS data structures cannot be completely initialized within one Calc3 icon. Each element of a PLS must be configured individually.

Typically, PLS configurations are first created using the PLS Tool under ***Commission ⇒ PLS***. Afterwards, the Calc3 icon can be used to modify different elements in the configuration. The following three PLS types can have their elements written to using the Calc3 icon.

- Option Card PLS
- Control PLS
- Drive PLS

Option Card PLS

A user program can contain only one active Option Card PLS. The following tables list the elements of a Option Card PLS.

Master Elements

The following Master elements are available for modification using the Calc3 icon.

Syntax	Description	Example	Comment	Parameter
PLSP[1].o***y***	Phase Offset	PLSP[1].o1=20	add an offset of 20 to PLS Master 1	C-0-2943
y = Master range 1-8				

Switch Elements

The following switch elements are available for modification using the Calc3 icon.

Syntax	Description	Example	Comment	Parameter
PLSP[1].on***y***	On Position	PLSP[1].on3=40	switch 3 On position is 40	C-0-2920
PLSP[1].off***y***	Off Position	PLSP[1].on3=60	switch 3 Off position is 60	C-0-2921
PLSP[1].out***y***	Assigned Output	PLSP[1].out3=2	switch 3 is assigned to output 2	C-0-2922
y = switch range 1- 96				

Output Elements

The following Output elements are available for modification using the Calc3 icon.

Syntax	Description	Example	Comment	Parameter
PLSP[1].lt***y***	Lead Time	PLSP[1].lt2=500	add a lead time of 500µs to output 2	C-0-2931
PLSP[1].lg***y***	Lag Time	PLSP[1].lg2=500	add a lag time of 500µs to output 2	C-0-2932
PLSP[1].ss***y***	PT (Time Duration)	PLSP[1].ss2=1000	maintain output 2 On for 1000µs	C-0-2933
PLSP[1].h***y***	Hysteresis	PLSP[1].h2=0.1	add a Hysteresis of 0.1 to the output 2	C-0-2936
PLSP[1].md***y***	Mode	PLSP[1].md2=0	set mode of output 2 to Lag Time (0=Lag Time, 1=PT)	C-0-2934
PLSP[1].dr***y***	Direction	PLSP[1].dr2=0	set direction of output 2 to positive (0=pos, 1=neg, 2=pos/neg)	C-0-2935
y = output range 1-32				

Control PLS

VisualMotion Control PLS are stored with the user program as a separate table, and can be assigned to an ELS Master, ELS Group or Real Master. A user program contains two active Control PLS. The following table lists the elements of a control PLS that can be easily modified using the Calc3 icon.

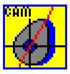
Syntax	Description	Variable Range	Example	Comment
PLS[***x***].a	Number	1-40 when t=3 or 4, Drive No. 1-6 when t=5, ELS Master 1-8 when t=6, ELS Group	PLS[1].a=2	set drive number of PLS 1 to 2
PLS[***x***].o	Phase Offset		PLS[1].o=20	add an offset of 20 to PLS 1
PLS[***x***].r	Output Register		PLS[1].r=70	set output register of PLS 1 to 70
PLS[***x***].m	Mask Register		PLS[1].m=72	set mask register of PLS 1 to 72
PLS[***x***].t	PLS Master Type	3 = Drive's primary feedback 4 = Drive's secondary feedback 5 = ELS Master 6 = ELS Group	PLS[1].t=3	set master type of PLS 1 to 3 (drive's primary feedback)
PLS[***x***].on1-on16	On Position		PLS[1].on1=10	switch 1 On position is 10
PLS[***x***].off1-off16	Off Position		PLS[1].off1=20	switch 1 Off position is 20
PLS[***x***].lt1-lt16	Lead Time (ms)		PLS[1].lt1=5	switch 1 lead time is 5
x = PLS number 1-2				

Drive PLS

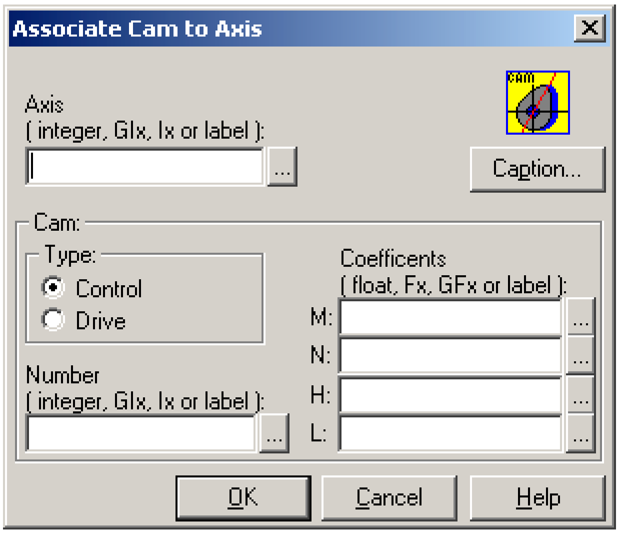
VisualMotion 11 supports drive based PLS configurations for Rexroth Diax 04 and Rexroth EcoDrive 03 digital drives. The drive based PLS is stored on the drive in list parameters. The elements of this list cannot be modified individually. Modifying one element of the list requires sending the entire list over the service channel.

Syntax	Description	Example	Comment	Parameter
PLSD[***x***].r	Output Register	PLSD[1].r=72	set output register of drive 1 to 72	A-0-0009
PLSD[***x***].t	PLS Master Type	PLSD[1].t=1	set master type of drive 1 to 1 (0=disable, 1=motor encoder, 2=external encoder)	P-0-0131
PLSD[***x***].on1-on16	On Position	PLSD[1].on1=10	switch 1 On position for drive 1 is 10	P-0-0132
PLSD[***x***].off1-off16	Off Position	PLSD[1].off1=20	switch 1 Off position for drive 1 is 20	P-0-0133
PLSD[***x***].lt1-lt16	Lead Time (ms)	PLSD[1].lt1=5	switch 1 lead time for drive 1 is 5	P-0-0134
x = drive number range 1-40				

CAM



The CAM icon is used to associate a CAM to a slave axis and to supply coefficients for using the CAM. To work properly, the axis must be configured as an ELS slave and synchronized to a master. The CAM can be drive or control resident.



CAM_Icon.tif

Fig. 14-20: CAM Icon Setup

Axis: Specifies which axis is associated with the CAM. The entry may be an integer, global integer variable (GIx), program integer variable (Ix) or an equivalent label.

CAM Number: Specifies which CAM table to use. The entry may be an integer, global integer variable (GIx), program integer variable (Ix) or an equivalent label (Range 1 to 37 for control CAMs and 1 to 2 for drive CAMs.)

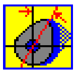
Coefficients: M, N, H and L may be entered as floats, global float variables (GFx), program float variables (Fx), or equivalent labels (control CAMs only).

Note: The H coefficient of a CAM verifies based on the target firmware:

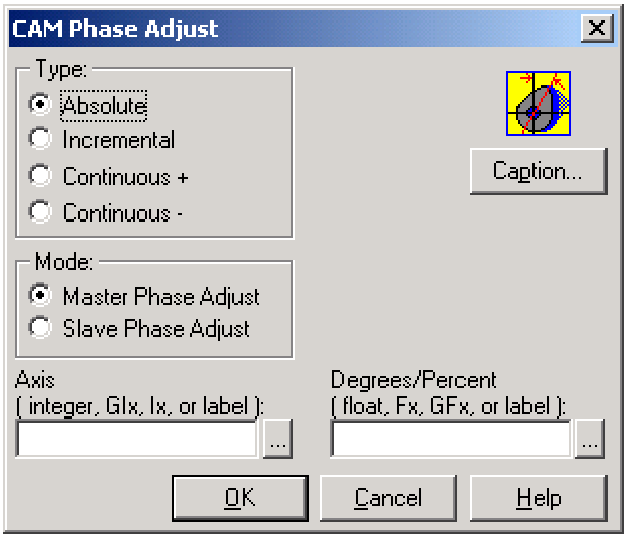
For GPP 7/ 8 the H value is scaled in units.

For G*P 9, G*P 10, G*P 11 or Drive, the H value is scaled in percent.

CAMAdj



This icon selects and starts a phase adjust. There are two phase offset values for a CAM axis: a **Master Phase Adjust**, and a **Slave Phase Adjust**. The **Master Phase Adjust** shifts the position in the CAM table relative to the master position. The **Slave Phase Adjust** shifts the position of the slave axis. Since it is not related to the shape of the CAM, the **Slave Phase Adjust** is not multiplied by any of the CAM factors.



CAMAdj_Icon.tif

Fig. 14-21: CAMAdj Icon Setup

The **Master Phase Adjust** adds the phase adjust value to the sampled ELS master position before the slave position is referenced in the CAM table. For control CAMs, the **Master Phase Adjust** is written to axis parameter A-0-0151. For drive based CAMs, the **Master Phase Adjust** is written to drive parameter P-0-0061.

The **Slave Phase Adjust** adds the phase adjust value to the output position of the CAM table. For control CAMs, the **Slave Phase Adjust** is written to axis parameter A-0-0161. For drive based CAMs, the **Slave Phase Adjust** is written to axis parameter A-0-0151.

Type selects the method of adjustment:

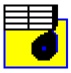
If **absolute**, the **degrees or percent** edit field is the new offset.

If **incremental**, the **degrees** edit field is added to the current offset. If in phase mode and sum exceeds 360, it rolls over. If in velocity mode the sum is limited to -100 or +300 percent.

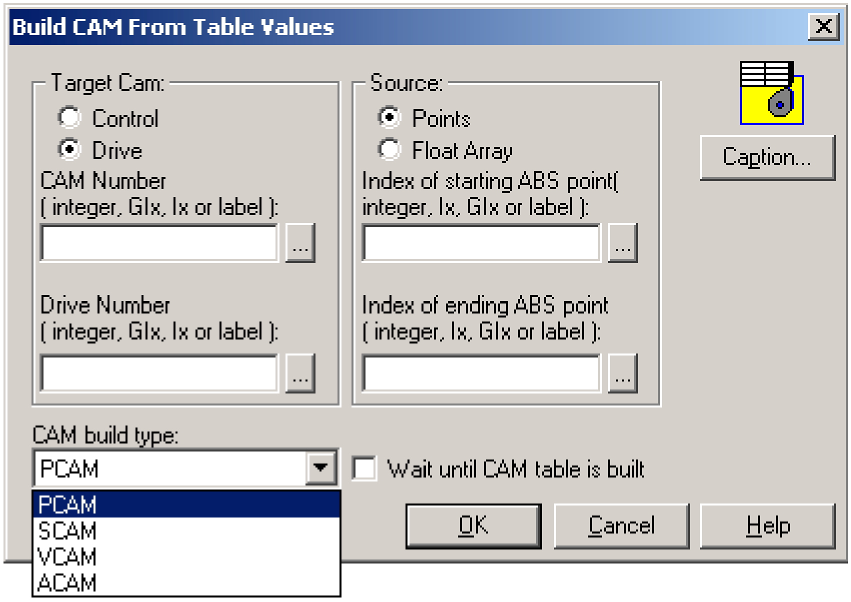
If **continuous +** or **continuous -** (phase sync mode only), a velocity dependent amount is added to the degree offset.

The control can perform only one phase adjust at a time. If two different phase adjust icons are used the second one will have no effect until the previous phase adjust is complete. Bit 4 in the axis status register is set to (0) when a phase offset is in progress and (1) if the phase offset is complete.

CamBld2



The control can build a CAM on-line based on a set of input positions. Program instructions use the control's ABS point table or float array and an internal utility to build an internal control CAM which is stored on the Control or Drive as the CAM number indicated.



CamBld2_Icon.tif

Fig. 14-22: CamBld2 Icon Setup

Target Cam:

From the Target Cam section, the user selects the location to store the control CAM. When **Control** is selected, the user can only specify the CAM number. When **Drive** is selected, the user can specify the CAM number and the Drive number to where the CAM will be stored.

CAM Number

This number indicates the CAM table number to build. The entry may be an integer, global integer variable (GIx), program integer variable (Ix) or an equivalent label (range 1 to 37 for control CAMs and 1 to 2 for drive CAMs). The CAM cannot be in use when this icon is executed.

Drive Number

This number (drive CAMs only) indicates which drive to store the CAM. The entry may be an integer, global integer variable (GIx), program integer variable (Ix) or an equivalent label.

CAM Build Type:

The **PCAM** build type accepts input in the form of a table of target positions. The **VCAM** build type accepts a velocity profile as input. The **ACAM** build type accepts an acceleration profile as input and outputs a normalized profile. When **SCAM** is selected, the control builds the CAM by connecting the points with third order splines. A minimum of 5 and a maximum of 200 user defined points are allowed. The X elements of the point table are the master positions, and the Y elements are the corresponding slave positions. All GPP9 CAM profiles are normalized.

Source

The user selects the source of the control CAM. When selecting **Points**, the user must specify an *Index of starting ABS point* and an *Index of the ending point*.

ABS Starting Point

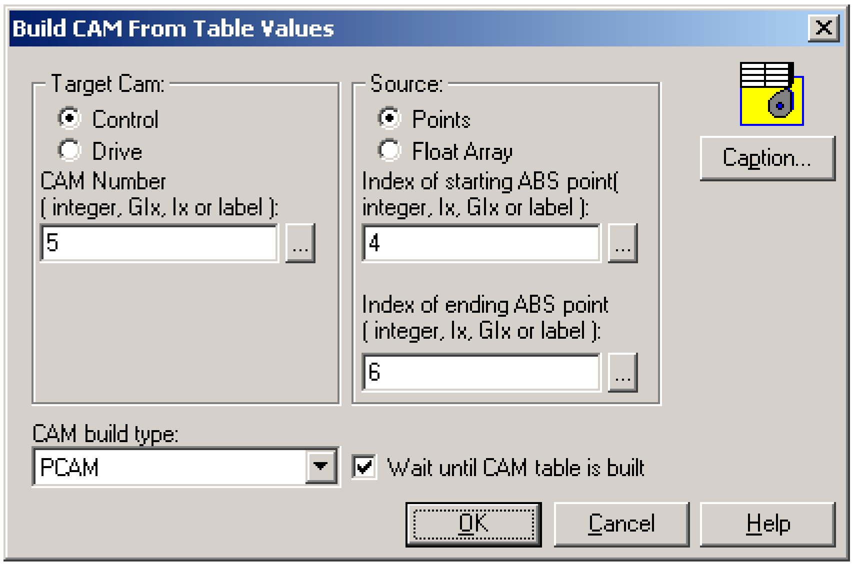
Specifies the starting ABS point number used for CAM generation. The entry may be an integer, global integer variable (GIx), program integer variable (Ix), or an equivalent label.

ABS Ending Point

Specifies the ending ABS point number used for CAM generation. The entry may be an integer, global integer variable (GIx), program integer variable (Ix), or an equivalent label.

Points Example:

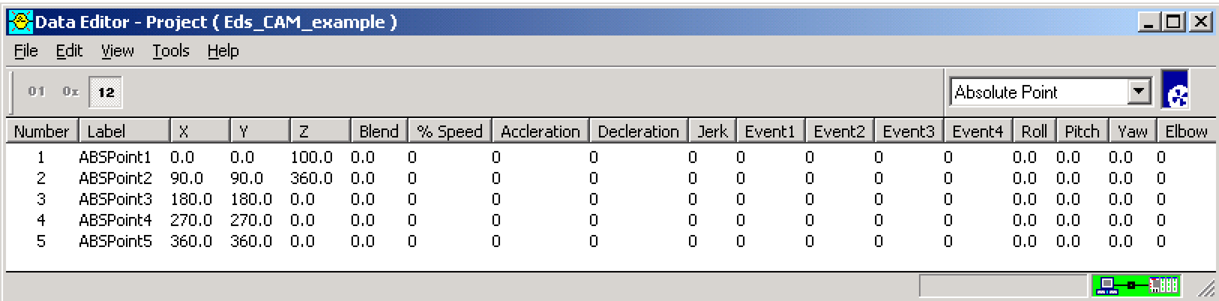
This example sets up a 3 points table that contains an x and y position for both a master and slave, starting at ABS point 4 and ending at ABS point 6.



CamBld2_Icon_Points.tif

Fig. 14-23: CamBld2 Icon Points Configuration

Once compiled and downloaded to the control, the user can write to the points table by selecting ***Data*** ⇒ ***Points*** and modifying the points directly, or within the user program using the Calc3 icon.



Number	Label	X	Y	Z	Blend	% Speed	Accleration	Decleration	Jerk	Event1	Event2	Event3	Event4	Roll	Pitch	Yaw	Elbow
1	ABSPoint1	0.0	0.0	100.0	0.0	0	0	0	0	0	0	0	0	0.0	0.0	0.0	0
2	ABSPoint2	90.0	90.0	360.0	0.0	0	0	0	0	0	0	0	0	0.0	0.0	0.0	0
3	ABSPoint3	180.0	180.0	0.0	0.0	0	0	0	0	0	0	0	0	0.0	0.0	0.0	0
4	ABSPoint4	270.0	270.0	0.0	0.0	0	0	0	0	0	0	0	0	0.0	0.0	0.0	0
5	ABSPoint5	360.0	360.0	0.0	0.0	0	0	0	0	0	0	0	0	0.0	0.0	0.0	0

Points.tif

Fig. 14-24: Points Table

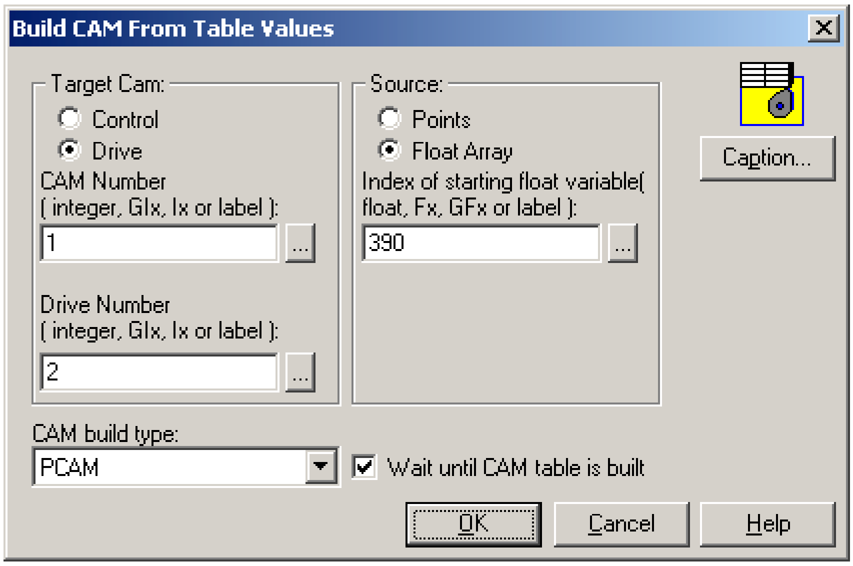
No.	X	Y	Z	Jk (Jerk)
004	First master position	First slave position	Slave Scaling value	Shaping Factor, 0 –100%
005	Second master position	Second slave position	Slave scaling position	not applicable
006	Last master position	Last slave position	not applicable	not applicable

Table 14-7: Example Points Table Description

When selecting **Float Array**, the user must specify the *Index of starting float variable*. This number identifies the first float variable number that will be used to store the data for the float array table.

Float Array Example:

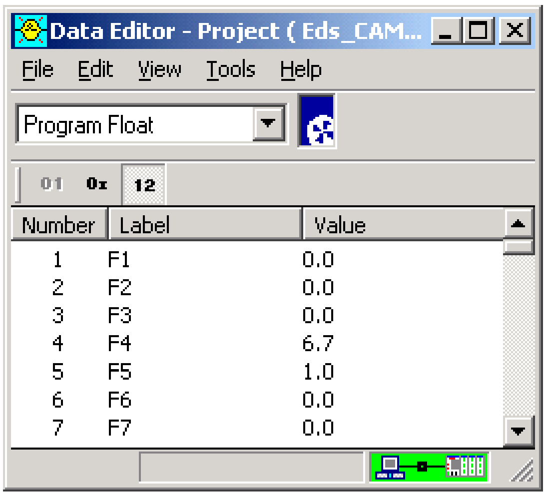
This example sets up a float array table that contains a count of 3 equally spaced CAM values, starting at float variable 390.



CamBld2__Float.tif

Fig. 14-25: CamBld2 Icon Setup for Float Array

Once compiled and downloaded to the control, the user can write to the float variables by selecting ***Data*** ⇒ ***Variables*** and modifying the variables directly, or within the user program using the Calc3 icon.



Variables.tif

Fig. 14-26: Variables Table

Float Variable	Description
F1	Number of equally spaced CAM values
F2	Slave scaling value
F3	Shaping factor, 0 – 100%
F4	Slave scaling position
F5	1st CAM value
F6	2nd CAM value
F7	3rd CAM value

Table 14-8: Example Float Table Description

General Information

The ABS point elements may be changed within the program by using CALC statements. Changes in the point table do not affect the CAM until the CamBld2 icon is executed. It is necessary to size the point table at compile-time to allow enough points for the profiles that will be needed. The x value of the first point must be 0, and the x value of the last point must be 360.

The CAM generation may take one second or longer. The ‘wait’ checkbox can be cleared to exit this icon immediately and keep executing instructions while the CAM is being built. The branch icon can be used to check if a CAM is ready for activation. If the ‘wait’ checkbox is checked, the program flow will be stopped in this icon until the CAM is ready for activation.

The CamBld2 icon can be used to store a CAM to an inactive location on the control or drive. After the CAM has been built, the CAM icon can select it for an axis.

Because the CAM is stored in nonvolatile memory on the control or the drive, it is not necessary to execute this command each time through the program. A flag variable can be set and checked the next time through the program to avoid long delays when starting the program. For on-line changes, a register bit or variable should be checked each time through the program loop to avoid continually generating the CAM, which consumes control resources and can slow down the program.

Control CAMs enter the control as text files in CSV format, the kind most spreadsheets generate. Drive CAMs enter a single column text file indicating the slave position.

If the CSV file for control CAMs contains less than 1024 points, an algorithm within the control fills in the missing points. As a rule, a CSV CAM file should contain at least 200 points, anything less than that does not sufficiently define the CAM and unexpected results may occur. Drive CAMs must contain exactly 1024 points.

Errors at runtime

The selected CAM is currently active for any axis.

The point range exceeds the bounds of the point table.

Less than two points are defined.

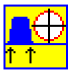
The CAM number is not valid (out of range or drive is not configured).

An error occurred when sending the CAM to the drive.

When using PCAM option, and the first x position isn’t 0 and the last x position isn’t 360.

The x position exceeds the modulo of the master.

CAM Indexer

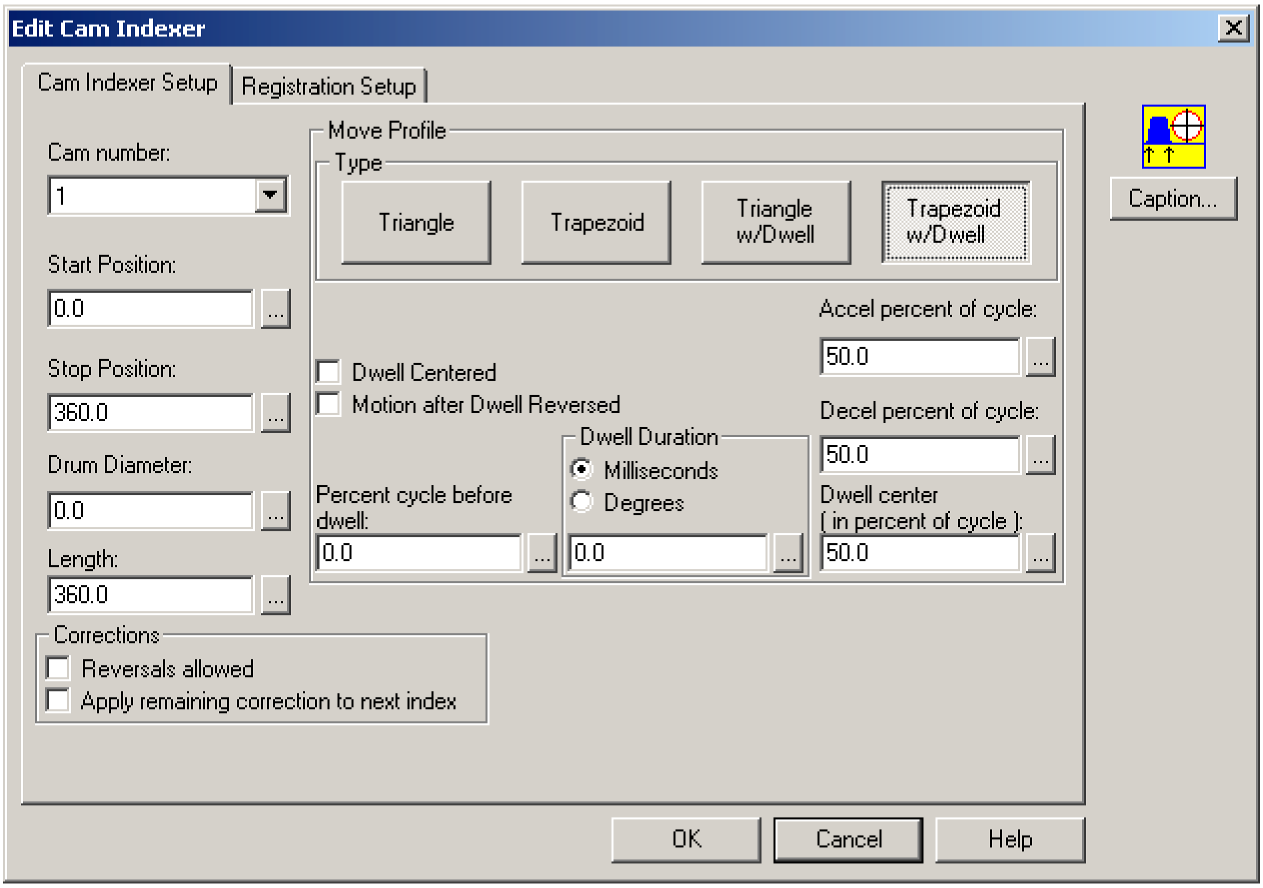


A CAM Indexer configuration must first be declared under ***Setup* ⇒ *Process* ⇒ *CAM Indexer***. Afterwards, an CAM Index icon can be placed in the initialization task to modify an existing CAM Indexer setup.

The following features are supported:

- A maximum of 4 unique CAM Indexers are supported in a project
- Axes associated with a CAM Indexer must be of type control CAM

Note: A unique CAM Indexer number can only be assigned once in a VisualMotion project.



CAM Indexer.tif

Fig. 14-27: CAM Indexer Setup

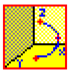
Cam Indexer Setup

Only configured CAM Indexers can be selected from the Cam Number field. For detailed descriptions of all the available settings, refer to section 4.6, *CAM Indexer Setup*, in volume 1.

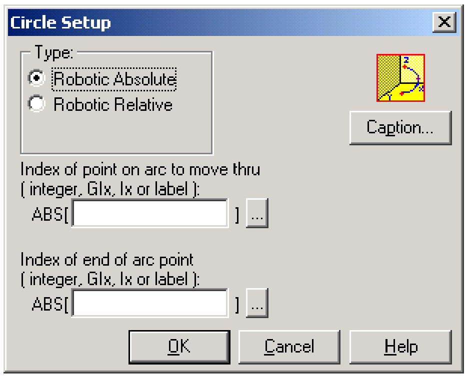
Registration Setup

The **Registration Setup** tab is only displayed when registration is enabled from the *Declaration* window under Setup. Refer to section 4.6, *CAM Indexer Setup*, in volume 1 for details. For specific information on Registration functionality, refer to section 8.4, *CAM Indexer*, in volume 1.

Circle



The Circle icon is used for multi-axis coordinated circular interpolation, moving in a circular arc from one point in 3D space to a second point in 3D space. The plane of the arc is two-dimensional and the plane may have any orientation in three-dimensional space. Placing a Circle icon on a Task or Subroutine workspace opens a *Coordinated Circle Setup* window. The Go Icon is not required with Circle function. When executed, the motion will immediately be sent to the path planner. Stepping to the next icon in the program will take place immediately.



Circle_Icon.tif

Fig. 14-28: Circle Setup

Robotic Move

Move in a circular arc from the current point in space past the “thru” point to the end point. The plane of the arc is two-dimensional and may have any orientation in three-dimensional space.

A circular move may be Absolute or Relative and is defined by three points on the circle. These points must have been previously defined in the absolute and/or relative point tables. Points are identified by a point table ID number or equivalent label. Indirect point table entries may be used by specifying an integer variable, global or program (GIx, Ix) or equivalent label.

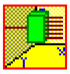
An absolute circular move begins from the endpoint of the previous segment (or current position if the system is halted), moves through an absolute point, and terminates at another absolute point.

A relative circular move is similar; however, the intermediate and endpoints are defined as relative offsets from the starting point. The relative circular move begins at the endpoint of the previous segment (or current position if the system is halted), moves through the first relative offset, and terminates at the second relative offset.

Coordinated Motion Event Triggers

Event triggers for coordinated motion are associated with a points table. For specific information on *Coordinated Motion Events,* refer to section 5.2, Events, in volume 1.

CoordArt2



A Coordinated Articulation configuration must first be declared under ***Setup ⇒ Processes ⇒ ELS ⇒ Coordinated Articulation***. Afterwards, a CoordArt icon can be placed in the initialization task to modify an existing Coordinated Articulation setup.

Refer to Coordinated Articulation Setup in chapter 4 of volume 1 for details. For specific information on Coordinated Articulation functionality, refer to section 6.2 in volume 1.

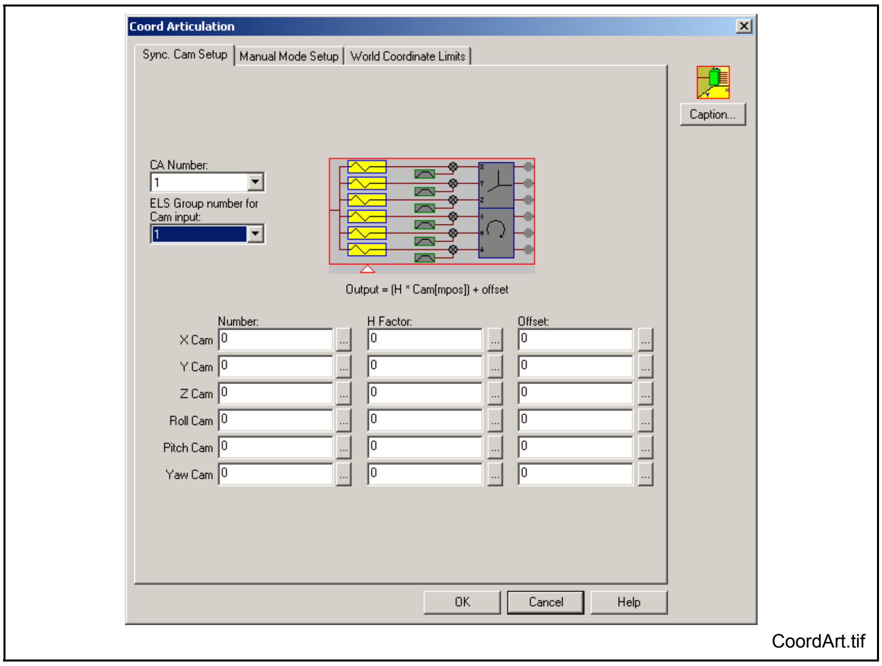


Fig. 14-29: Coordinated Articulation

Sync. Cam Setup

The Sync. *CAM Setup* window is used to configure the H Factor and Offset for the selected ELS Group number. Only Coordinated Articulation configurations that were declared under Setup can be selected. Up to 4 Coordinated Articulation configurations can exist in a project at one time.

Note: Only ELS Groups setup under ***Setup ⇒ Processes ⇒ ELS ⇒ ELS Groups*** will appear as a valid CAM input.

Manual Mode Setup

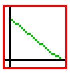
The *Manual Mode Setup* window is used to set the acceleration, deceleration and velocity that will be used when moving Coordinated Articulated axes in local mode.

World Coordinated Limits

The *World Coordinated Limits* window is used to set the minimum and maximum boundaries that the system will use for the X, Y, Z, Roll, Pitch and Yaw input values.

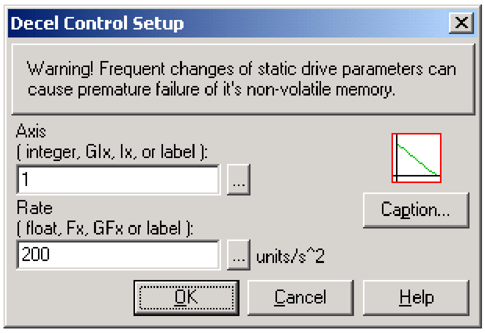
Note: It is strongly recommended that the world coordinated limits be setup to ensure protection of all hardware.

Decel



The Decel icon is used to set or change the deceleration of a single non-coordinated axis. The deceleration function is only available in the following drive firmwares:

- Rexroth EcoDrive 03 (SGP03 firmware)
- Rexroth IndraDrive (all firmware types)



Decel_Icon.tif

Fig. 14-30: Decel Icon Setup

Axis indicates the axis to decelerate. The axis may be entered as an integer constant, integer variable, global integer variable or an equivalent label.

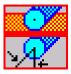
Rate specifies the deceleration rate in units per second squared. The entry may be a float constant, variable, global variable or an equivalent label. The variable must be greater than 0.

Units: Axis Definition

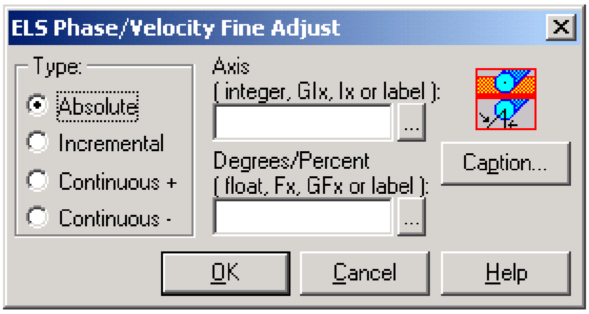
Single Axis Linear - inches	inches/s^2
Single Axis Linear - mm	mm/s^2
Single Axis Rotary	rads/s^2
Velocity Mode	rads/s^2

Clicking the browse button to the right of each data field opens the VM Data Table.

ELSAdj1



The ELSAdj1 icon is used to adjust the velocity or phase of an ELS configured axis to compensate for mechanical variations between master and slave axes. The axis may be specified by an integer constant, variable, global variable or an equivalent label. Clicking the browse button to the right of any field opens the VM Data Table.



ELSAdj_Icon.tif

Fig. 14-31: ELSAdj Setup

Type

This section selects the method of adjustment:

- **absolute -** the **degrees or percent** edit field is the new offset.
- **incremental -** the **degrees or percent** edit field is added to the current offset. If in phase mode and sum exceeds 360, it rolls over. If in velocity mode, the sum is limited to -100 or +300 percent.
- **continuous +** or **continuous -** (phase sync mode only), a velocity dependent amount is added to the degree offset.

For phase synchronization the resulting slave phase is:

$Ø_s = (Ø_m * (K_s/K_m)) + \text{adjust}$

For velocity synchronization the resulting slave velocity is:

$V_s = V_m * ((1 + \text{adjust}) * (K_s/K_m)$

Where:

$Ø_m$ = master axis phase

V_m = master axis velocity

K_s = slave axis master/slave ratio turns value

K_m = master axis master/slave ratio turns value

adjust = a value in the range of -100% to +300% for ratio, or 0° to +360° for phase.

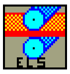
Axis

This field indicates the axis to be adjusted. The entry may be an integer, global integer variable (GI1- GI256), program integer variable (Ix), or an equivalent label.

Degrees or Percent

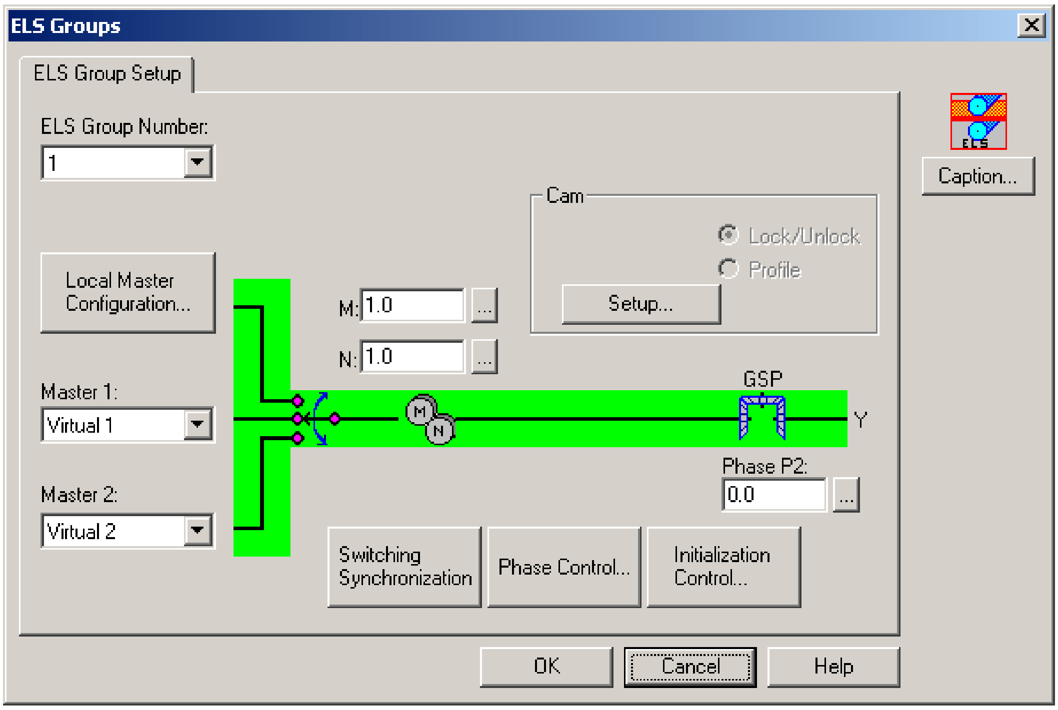
Degrees (phase) or percent (velocity) - When enabled this allows fine offset to be added. This entry may be a float, global float variable (GF1-GF256), program float variable (Fx), or equivalent label. Valid adjustment ranges are 0 to 360 degrees or -100% to +300%.

ELSGrp3



An ELS Group configuration must first be declared under ***Setup*** ⇒ ***Processes*** ⇒ ***ELS*** ⇒ ***ELS Groups***. Afterwards, an ELSGrp icon can be placed in the initialization task to modify an existing ELS Group setup.

VisualMotion 11 supports up to eight ELS Groups in a G*P 11 project.



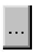
ELSGrp_Icon.tif

Fig. 14-32: ELS Group Setup

ELS Group Setup

The following data can be modified from the ELSGrp icon:

- ELS Group Number Selection
- Local Master Configuration button
- Master Selection
- CAM Selection
- Switching Synchronization button
- Phase Control button
- Initialization Control

Clicking the browse  button to the right of M, N, H, Phase P1 or Phase P2 allows selection of a variable from the VM Data Table.

Refer to ELS Group Setup in chapter 4 of volume 1 for details.

ELS Group Number: Select an ELS Group number from the drop-down list. Only ELS Groups configured in the Setup will be available for selection.

Local Master Configuration The *Local Master Configuration* is used to set the Initial, Maximum and Positioning values for the ELS Group. These values are used when the ELS Group is switched to local mode for jogging purposes. For specific information on how to setup the Local Master Configuration, refer to section 6.1 in volume 1.

Master Selection

The *Master 1* and *Master 2* drop-down list is used to set the ELS System Master type that will be used for the current ELS Group. The available master types are dependent on the *ELS Masters* configured under **Setup**. Refer to section 4.8, *System Master Setup*, in volume 1.

CAM Section

The CAM section is used to assigned a configured control CAM or CAM Indexer to the ELS Group.

Synchronization Setup

Synchronization setup is used to configured the method that will be used when the ELS Group's master input is switched between the two available masters. For specific information on how to setup synchronization, refer to section 6.1 in volume 1.

Phase Control

Phase control is used to select the phase adjustment type that will be used for the master and slave phase adjust.

The available type are as follows:

- Trapezoidal profile
- Immediate

For specific information on how to setup phase control, refer to section 6.1 in volume 1.

Initialization Control

Initialization control is used to select the ELS group master operation options. Refer to ELS Group Setup in chapter 4 of volume 1 for details.

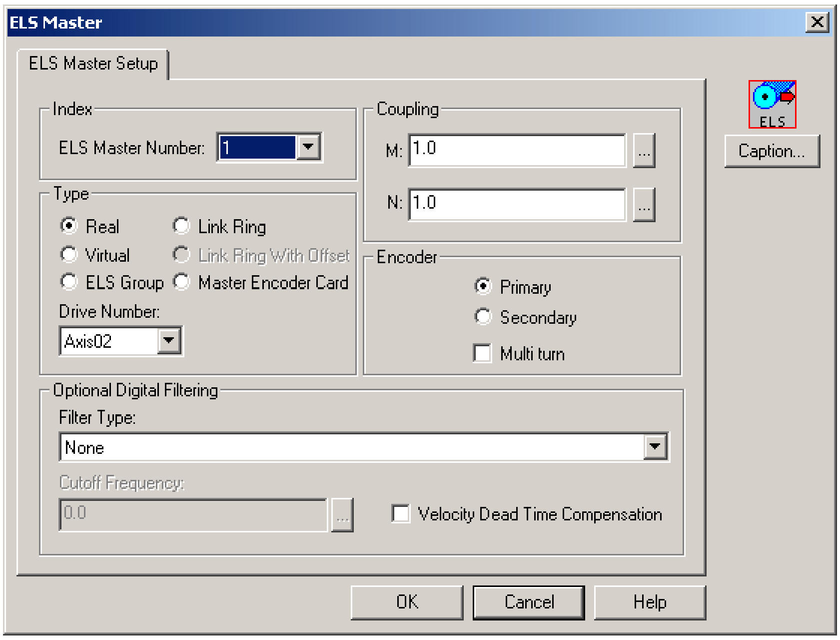
ELSMstr2

An ELS System Master configuration must first be declared under ***Setup ⇒ Processes ⇒ ELS ⇒ System Masters***. Afterwards, an ELSMstr2 icon can be placed in the initialization task to modify an existing ELS Master setup. VisualMotion 11 supports up to six ELS Masters in a G*P 11 project.

The ELSMstr2 icon allows the programmer to change the configuration of an existing ELS Master setup or place a new ELS Master setup as part of the runtime program flow. This flexibility allows the programmer to declare ELS System Masters based on specific program conditions.

Note: The ELSMstr2 icon has been changed to include only runtime initialized settings for G*P 11 projects. Pre-G*P 11 projects opened with VisualMotion Toolkit 11 and using a previous version of the ELSMstr icon which can still be compiled and downloaded without any additional modifications.

When placed in the Initialization task, the following setup window opens:



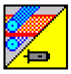
ELSMstr.tif

Fig. 14-33: ELS Master Assignment

Refer to *System Masters Setup* in section 4.8 of volume 1 for a complete description of the setting for the *ELS System Master Setup* window.

Rexroth
Bosch Group

ELSMode



The ELSMode icon is used to switch between Single Axis, Velocity and Sync to Master Modes. Once switched into Single axis mode, the axis may be positioned independent of any ELS master/slave relationship (e.g., jogged into position), then returned to the master/slave condition.

Note: When an Mode Change icon is used to change an axis from Single Axis mode to Sync to Master mode, a second Mode Change icon must be used if the user program is to encounter a single axis icon, such as Home.

The ELSMode icon can also be used to switch an axis that is configured for Single Axis Mode into Velocity Mode. Axis parameter **A-0-0180** must be set to 36 to place command velocity into the cyclic data telegram. Axis parameter **A-0-0004**, **bit 6** must be set to 1 to enable acceleration. The Sync to Master mode is then ignored.

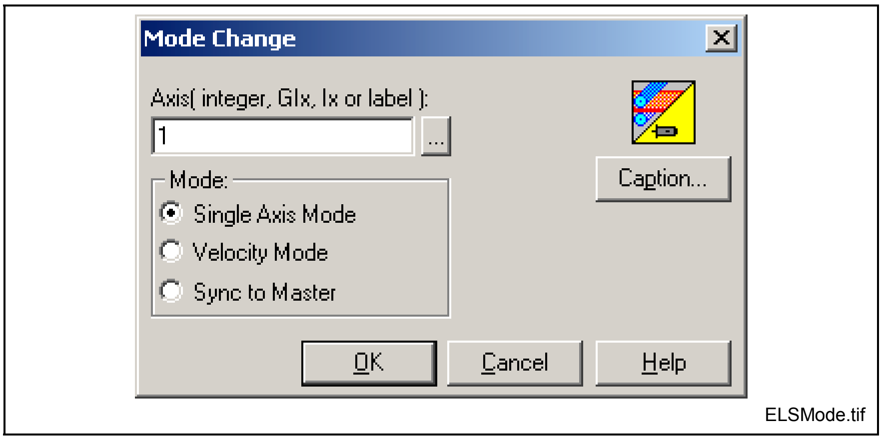


Fig. 14-34: ELS Mode Change

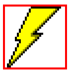
The axis is specified by an integer constant, variable, global variable or an equivalent label. Click the browse button to locate a data type from the VM Data Table. Entering “-1” in the axis box will send the mode change command to all ELS axes that were defined in the Task where the command is issued.

When an axis’ Sync type is configured for Phase or Drive CAM mode is switched into sync mode, a relative phase offset is automatically initialized between the slave and the master. The drive does not move into absolute synchronization with the master.

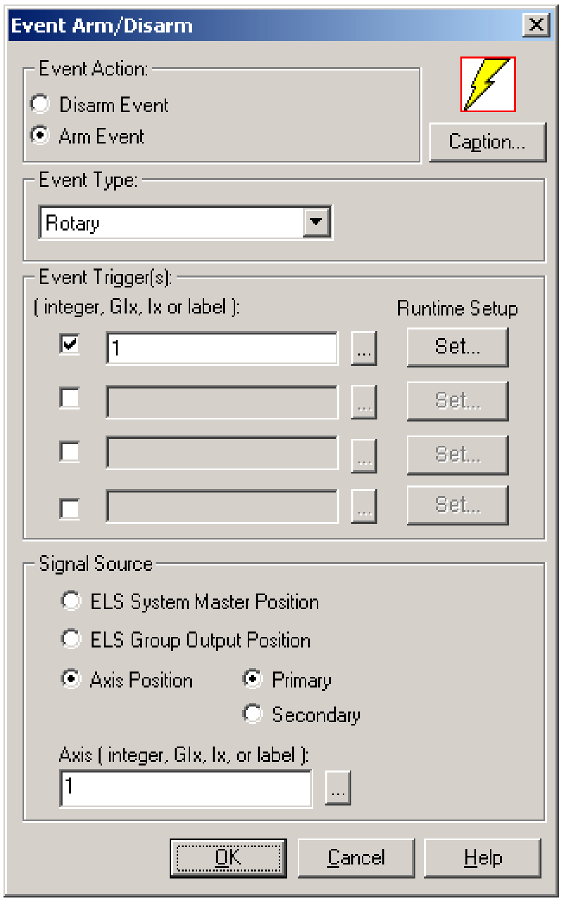
Care should be taken when switching an axis into synchronization with a moving master. Rexroth Diax 04 drives have the “ramp up and lock on” feature that assures smooth acceleration when synchronizing to a moving master.

When an axis is switched to single axis mode while the master is moving, it does not decel to a stop. It will stop at the last valid position command. To switch to single axis mode from following a moving master, first switch to velocity mode. The slave will continue moving at the last sampled master velocity (even in phase sync) and can be ramped down to a stop using the stop icons.

Event2



The Event2 icon allows the user to completely configure Events in project mode (Offline or Online) using a systematic process. Events are used to start an Event function (subroutine) when a specific condition (Event Trigger) is encountered in the program.



Event.tif

Fig. 14-35: Event Setup

Event Action:

Choose the event action to be perform by selecting one of two following radio buttons:

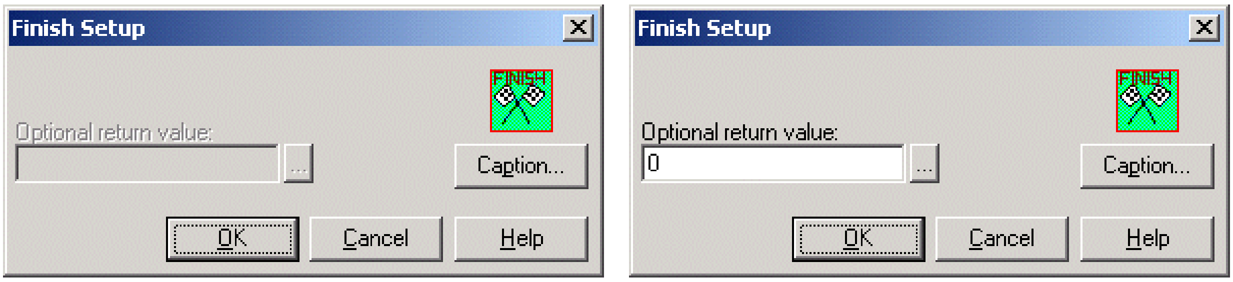
- **Disarm Event** - de-activates the specified Event trigger. If the event has already been made inactive, the icon has no effect.
- **Arm Event** – used to arm the 4 event types available from the Event Type drop-down list. Also used to when changing the argument of an event.

Note: A "Wait for Event Done" condition is performed by using the Wait icon.

For specific information on how to use the Event icon, refer to section 5.2 in volume 1.

Finish1

Each program task, subroutine and event function must end with a single Finish1 icon. Subroutines can return a single optional argument value to the calling function. The return argument may be a constant or a variable. A return argument is a convenient way to get position when using a common subroutine from more than one task. The **Optional return value** in the finish icon is only enabled from a finish icon in a subroutine. Task and event function cannot return an argument value. Refer to Optional Function Arguments (Sub1 icon) on page 14-81 for details.

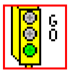


Finish icon in a subroutine.

Finish.tif

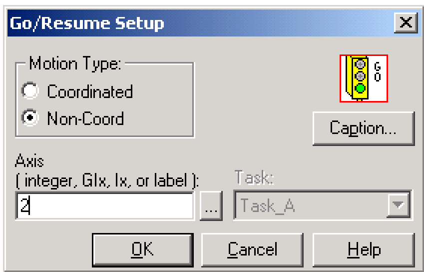
Fig. 14-36: Finish Setup

Go1



The Go1 icon is used to enable one or all non-coordinated axes used in any task. It also enables the associated position, velocity or servo loops. It should be placed before an associated Move2 icon. The Go1 icon can also be used to resume multi-axis coordinated motion that has been stopped.

Placing a Go1 icon opens the *Go Setup* window. The ***Motion Type*** radio button specifies the type of motion to start. Choosing **Non-Coord** as a **Motion Type** requires an entry in the **Axis** field specifying the control axis to start. The axis is specified by a valid integer constant, variable, global variable or an equivalent label. Specifying "-1" in the **Axis** field enables all axes in single axes or velocity mode assigned to the current task



Go.tif

Fig. 14-37: Go/Resume Setup

Choosing **Coordinated** as a **Motion Type** resumes motion to a coordinated axis halted by a **Coordinated Stop**. The task containing the halt coordinated axis must be selected from the Task drop-down list. Aborted coordinated motion should not be simply resumed.

Home

The Home icon is used to activate the drive's internal homing command for the specified axis in the **Axis to home** field. This is a single axis non-coordinated motion command. Placing a Home icon in a task or subroutine workspace automatically opens the *Homing Setup* window.

Note: The Home icon is used only for single-turn encoders. For multi-turn encoders, a move icon is used to move the axis to an absolute position, which will serve as the reference point for all moves.

Home.tif

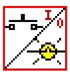
Fig. 14-38: Homing Setup

The axis to be home may be specified by a valid integer constant, variable (Ix), global variable (GIx), or an equivalent label. Click the browse button to select an axis from the VM Data Table.

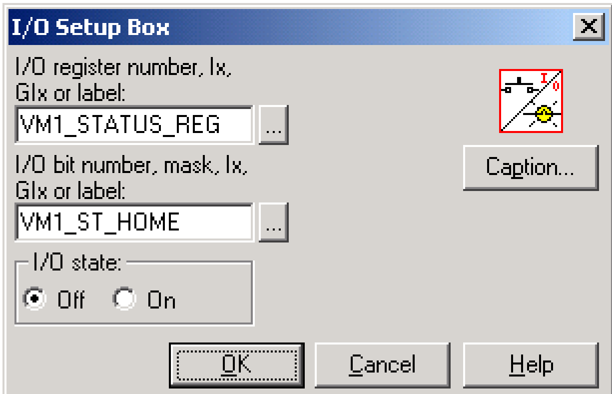
Once commanded to home, the drive homes the axis to the home position without further intervention from the control. Any errors in the drive's homing command are reported to the control. The program flow waits in this icon until the homing sequence is complete.

The Home icon uses the internal homing capability of the Rexroth's intelligent digital drive to perform the homing operation. For homing to occur, the homing parameters in the specified drive must have been setup before executing the Home icon. Refer to the relevant drive manual for details.

I_O (I/O Setup)



The I_O icon is used to control the state of I/O register bits. Placing an I_O icon in a task or subroutine workspace automatically opens an *I/O Setup* window.



IO_Setup.tif

Fig. 14-39: I/O Setup

The **I/O register name** field specifies the target register number. The entry can be a number, an integer variable, or a defined register label.

The **I/O Bit number** field specifies the bit or bits to be controlled and permits entry of an integer constant or equivalent label, as an "and" mask for the target register. A single bit is specified by a number, an integer variable, or a defined bit label. Multiple bits in a single register are specified in hexadecimal by a bit mask (e.g., 0x21 would specify bits 5 and 0).

Multiple bits can be changed with a single icon by entering an I/O bit mask that specifies more than one bit (e.g. 0x21 or a label equivalent to the desired mask). Clicking the browse button to the right of any field opens the VM Data Table.

The radio buttons for I/O state determine whether the target register bits, enabled by the I/O Bit mask, are cleared (logic zero, or off) or set (logic one, or on).

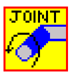
Clicking the browse button next to each field opens the VM Data Table which permits adding, editing or deleting register labels. The Register Labels window provides a scrolling list of default labels for the standard control system, axis, task, and digital drive I/O card control and status registers.

Join

A Join icon makes it possible to connect one line to another. VisualMotion icons have a maximum number of inputs; "join" overcomes this limitation by permitting many program flow paths to combine into a single path.

The Join icon is often the only method of completing a program flow, since VisualMotion cannot cross interconnecting lines. In addition, your program may require branching to several different calculations depending upon a certain condition. After the Calc3 icons you can use join icons to return to the main program flow before it enters the next icon.

Joint

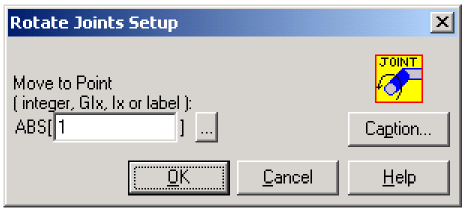


The Joint icon is used for point-to-point movement and joint (elbow) positioning, typical to robotic motion. It changes one or more motor angles from the current set J0 to a new set J1 defined by a absolute(ABS[x]) point. This instruction is only for coordinated motion whose kinematics supports joint angles. It can only be used with a six axis robot with a second frame of reference (more than just x, y, and z).

A Joint move is an absolute point-to-point move, with only the endpoint of the move specified. It is the most efficient type of move because the path calculated by the path planner is optimized to minimize time. A Joint move uses the axis' maximum acceleration and deceleration rates, while line and circle coordinated motion commands use Path Maximum percentages (defined in Task parameters) and Maximum Acceleration and Deceleration rates (defined in Axis parameters). Rate limiting is based on the most efficient axis limiting without violating the axis.

The actual path taken to the specified point is not defined and may assume whatever form the path planner requires; however, once programmed, the path can be repeated.

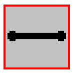
The destination is specified as an entry in the absolute point table as an integer constant, variable, global variable or an equivalent label.



Joint.tif

Fig. 14-40: Rotate Joints Setup

Line

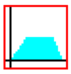


All icons in each task, subroutine and event function must be connected. The line icon is used to draw a line indicating program flow from one icon to another. Clicking on the beginning icon surrounds the icon with a box. Clicking on the ending icon automatically draws a line from the first to the second icon, with an arrowhead indicating the direction of program flow.

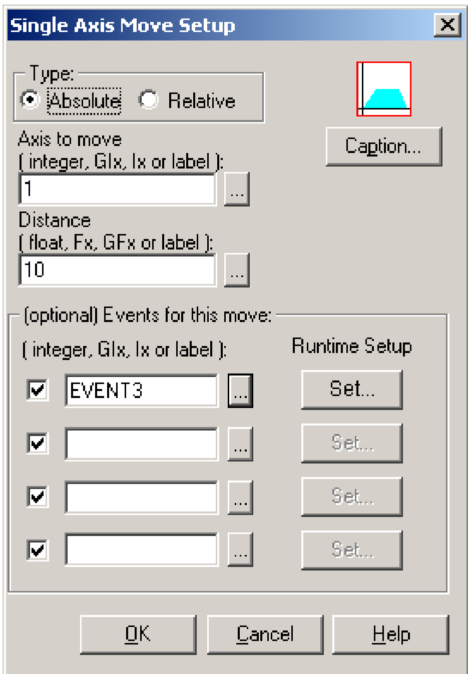
Under some circumstances, VisualMotion may be unable to route a line and displays a *Connection could not be made, try connecting adjacent blocks* window. Lines may be manually routed by clicking adjacent empty squares on the invisible workspace grid from the first icon to the second. A manually placed line may not cross another line; attempting to do so displays an error box.

A line connecting two icons may be deleted by using the Scissors icon located next to the Line icon. Position the Scissors over the line and click the left mouse button.

Move2



The Move2 icon is used to program movement on any single non-coordinated axis from any task. The Move2 icon initiates motion only if the axis has been enabled previously with a “GO” icon. Placing a Move2 icon on a task or subroutine workspace automatically displays a *Single Axis Move Setup* window.



Move.tif

Fig. 14-41: Single Axis Move Setup

The **Type** radio buttons specify either an incremental move distance added to the current position (Relative) or a move to an absolute position (Absolute).

The **Axis to move** field specifies the axis to move. It accepts an integer number, constant or variable.

The **Distance** field specifies the incremental move distance or absolute target position. It accepts a float number, constant or variable.

Single Axis Motion Events

Event triggers for single axis motion are associated a non-coordinated move. For specific information on *Single Axis Motion Events,* refer to section 5.2 in volume 1 of the *VisualMotion 11 Functional Description*.

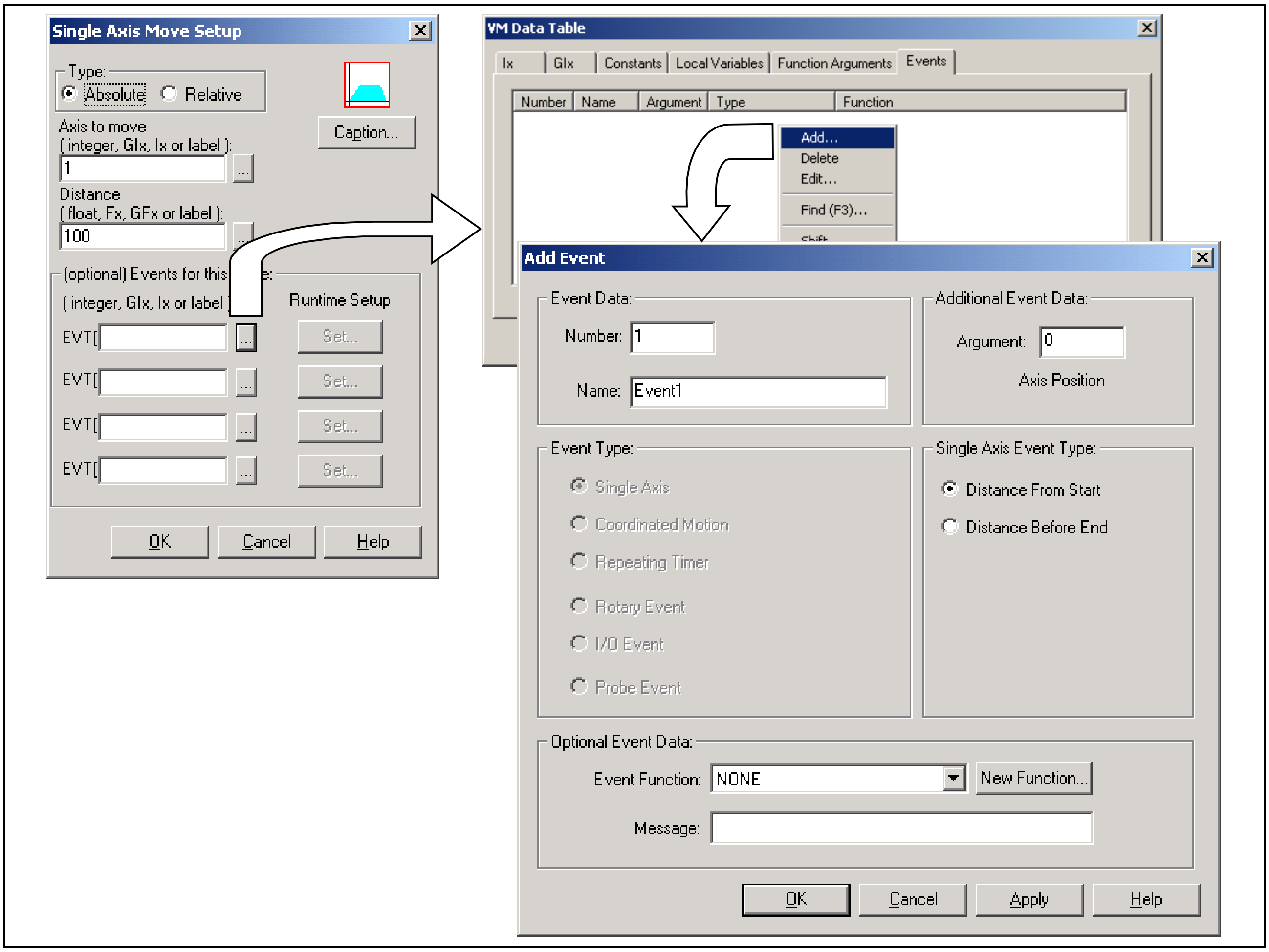


Fig. 14-42: Events in the Move Icon

Any event function assigned to an event trigger must be programmed Offline before running the program. The event types available for the “Move” icon are as follows:

- Single Axis Distance from Start
- Single Axis Distance from End

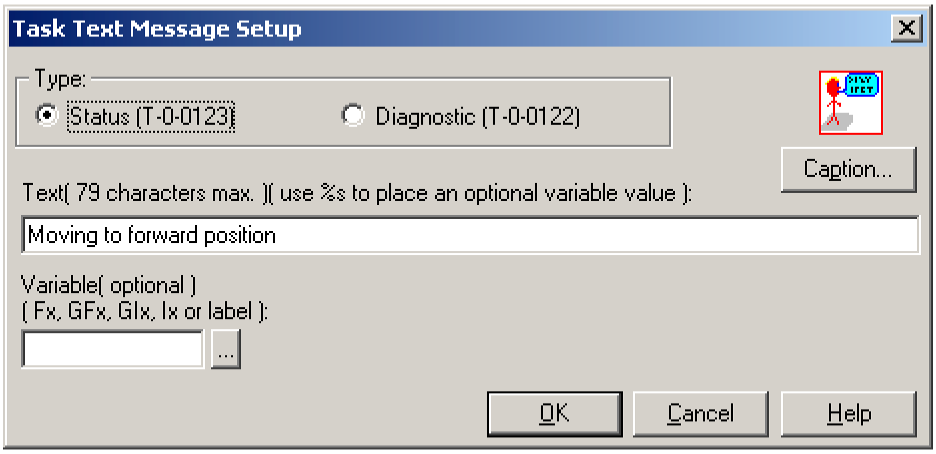
Msg1

The Message icon is used to select a status or diagnostic message (up to 79 characters) at a specified point in the program flow. Placing a Msg1 icon on a task or subroutine workspace automatically displays the *Task Text Message Setup* window.

Messages are used to inform a user about the current state of the program through Task Parameters T-0-0122 and T-0-0123, or through the teach pendant. Systems using Direct ASCII Communication may obtain messages through the RS-232 port. Messages may also be sent to the top line of the pendant and to the serial ports. Messages are available after the icon is executed in the program and remain in effect until another message of the same type is executed.

Status messages tell a user or machine operator something about the ongoing process. At runtime, the current status and diagnostic messages can be viewed by selecting ***Diagnostics*** ⇒ ***Tasks*** from VisualMotion's main menu.

Diagnostic messages are typically used to provide information about the current state of the system, e.g., "412 No drives were found on ring." If an error occurs during task execution, this diagnostic message is overwritten with an error message.



Message.tif

Fig. 14-43: Message Setup

An optional variable may also be displayed. This is useful for operator interface or debugging. A message may have one formatted variable in its string by using “%s” as a place holder for the variable entered in the **Variable (optional)** field.

Example

Message = "The current count is %s." *where %s equals variable I2 which has the label "current_count."*

The displayed variable may also have a corresponding label. The VM Data Table is opened when clicking on the browse button to the right of the **Variable (optional)** field. It is also a good idea to copy the message into the Icon *Caption and Comment* window by clicking on the ***Caption…*** button. This message will appear when the cursor is held over the Msg1 icon in the program.

Param1

The Param1 icon is used to transfer control (C, A, and T) and drive based (S and P) parameter values to variables and from variables to parameters. In addition to transferring parameter values, the Param1 icon also supports the execution and resetting of control and drive command parameters. Transferring parameters to variables and visa versa is the optimal method for performing calculations and logical operations on parameter values. A single Param1 icon can be configured for multiple parameter transfers and/or commands.

Note: The transferring of S and P drive parameters is only available when at least one axis is declared under Setup Axes within the Project Navigator.

The Param1 icon can be placed in any initialization and runtime task or subroutines, as well as event functions. Icons placed in the initialization task or initialization subroutines are executed, as part of the logical icon flow, during the control's phase 2 to phase 4 transition. Icons placed in any runtime task, subroutine, or event function are executed at runtime as part of the logical icon flow.

The following is a list of operations that can be performed using the Param1 icon:

- transfer a single variable to a standard or binary parameter
- transfer an ASCII string to a control text parameter
- transfer a variable array to a list parameter
- transfer a standard or binary parameter into a single variable
- transfer a list parameter into a variable array
- execute or reset a command parameter

Note: The **Param**, **ParamBit**, and **Command** icons have been removed from all icon palettes for G*P 11 projects. The functionality of these three icons has been combined and enhanced into the **Param1** icon.

The following parameter data types are supported by the Param1 icon:

- float and float array (list)
- integer and integer array (list)
- binary
- text (e.g., C-0-0142, C-0-0400)
- parameter commands

The following parameter data types are **not** supported:

- text list (e.g., C-0-2017)
- drive text parameters

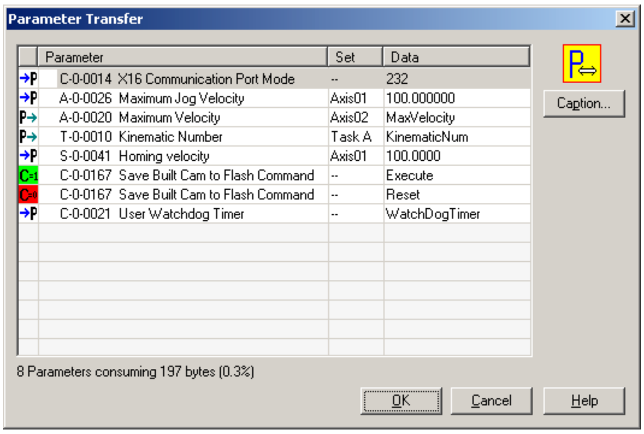
Note: Communication errors can occur when a program is activated that contains Param1 icons in the Initialization task that modify the X10 or X16 communication parameters. Refer to *Communication Errors After Program Activation* in section 10.3 under heading *Online Full Restore* for details.

Parameter Transfer List Icons and Table Structure

The following figure is an example of a configured Param1 icon. The *Parameter Transfer* window has four columns that identify the following items:

Item	Description
Operation Perform	The first column can displays the following: →P Writes a value to a parameter P→ Reads a value from a parameter or list parameter and write it to a variable(s) C Execute a command parameter (write) C Reset a command parameter (write)
Parameter	Parameter being read or written
Set	Parameter set containing the parameter being read or written.
Data	Value or variable label

Table 14-9: Parameter Transfer Window Data

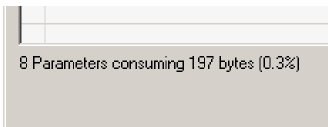


ConfigParam1.tif

Fig. 14-44: Configured Param1 Icon

Memory Usage

Each Param1 icon has a maximum memory size of 64 Kbytes. The parameter transfer wizard tracks and displays the current number of configured transfers and commands and displays the current memory consumed.



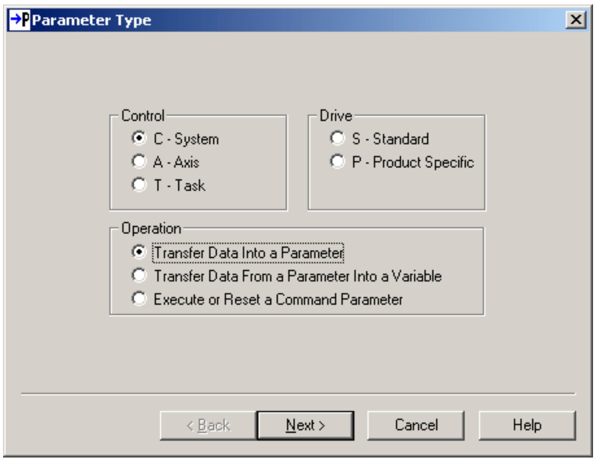
MemoryUsage.tif

Fig. 14-45: Memory Consumption

Rexroth
Bosch Group

Parameter Type

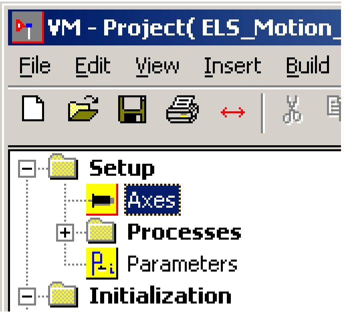
The *Parameter Type* window is used to specify the parameter type (control or drive) and operation that will be performed. The following figure displays the **Control** and **Drive** parameters as well as the **Operations** that are supported.



ParameterTransfer.tif

Fig. 14-46: Parameter Type Window

Note: Drive (S and P) parameters are grayed out when no axes are declared under Setup in the Project Navigator. This normally occurs when converting from a VM9 or VM10 project to a G*P11 project and the axes are defined within Axis icons. In order to use the Param1 icon to transfer S and P parameters in a G*P11 project, at least one axis must be declared under Setup Axes.



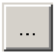
Once a parameter type and operation are selected, the window that follows depends on the parameter type selected. Refer to the following table for details:

Parameter Type Selected	Next Window
C - System	Parameter Number
A - Axis, T - Task, (S or P) - Drive	Parameter Set

Table 14-10: Windows following Parameter Type

Parameter Set

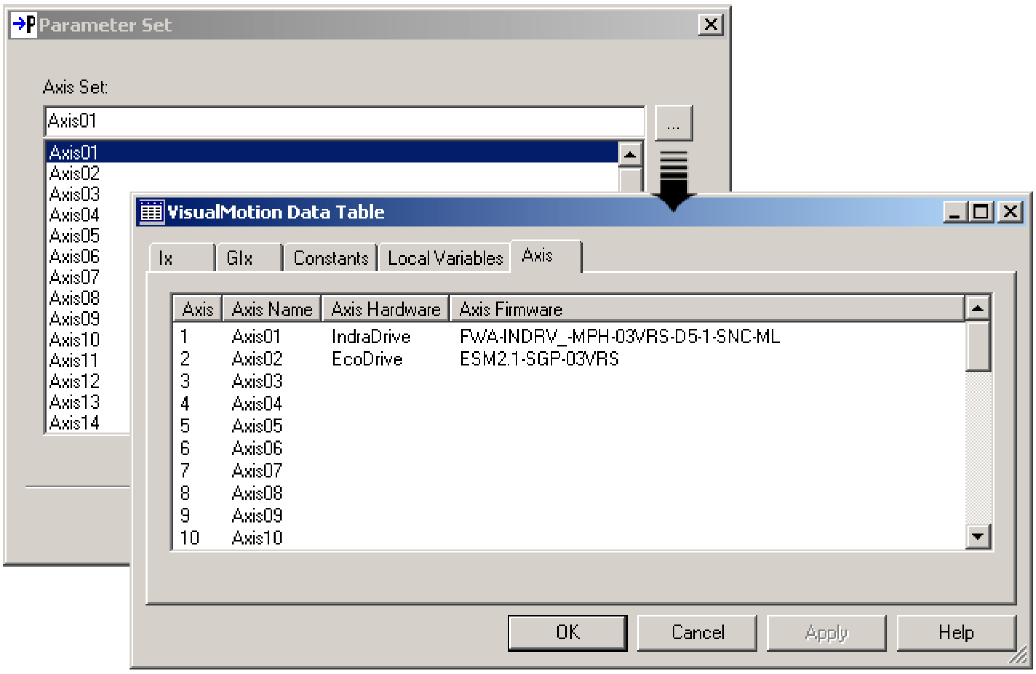
The *Parameter Set* window is displayed when an axis, task or drive parameter type is being configured. The user must specify which axis, task, or drive parameter set will be accessed.

Clicking the browse  button opens the VM Data Table displaying one or more of the following supported data types:

- Integers (Ix)
- Global Integers (GIx)
- Constants
- Local Variables
- Function Arguments
- Axis

The actual data types displayed in the VM Data Table depend on whether or not the parameter supports the data type.

Note: The *Parameter Set* window only allows for one parameter set to be selected from the list. To access multiple axis or task parameter sets, the user can use a variable and modify its value in a loop or subroutine.



ParamSetSelection.tif

Fig. 14-47: Selecting Axis Parameters

The number of items displayed in the *Parameter Set* window depends on the type selected. Refer to the following table for details:

Parameter Type	Available Number of Items
Axis	All 64 axis are available
Task	Task A-D are available
Drive	Only axes defined under Setup are available

Table 14-11: Parameter Set Selection

Clicking the ***Next >*** button, opens the Parameter Number window containing a list of valid parameters that can be configured for the items selected in the *Parameter Set* window.

Assigning a Variable to a Drive Parameter Set

When assigning a variable (integer) to a drive parameter set, VisualMotion will ask the user to define the Drive that will be assigned to the variable. Refer to the following figure for complete details.

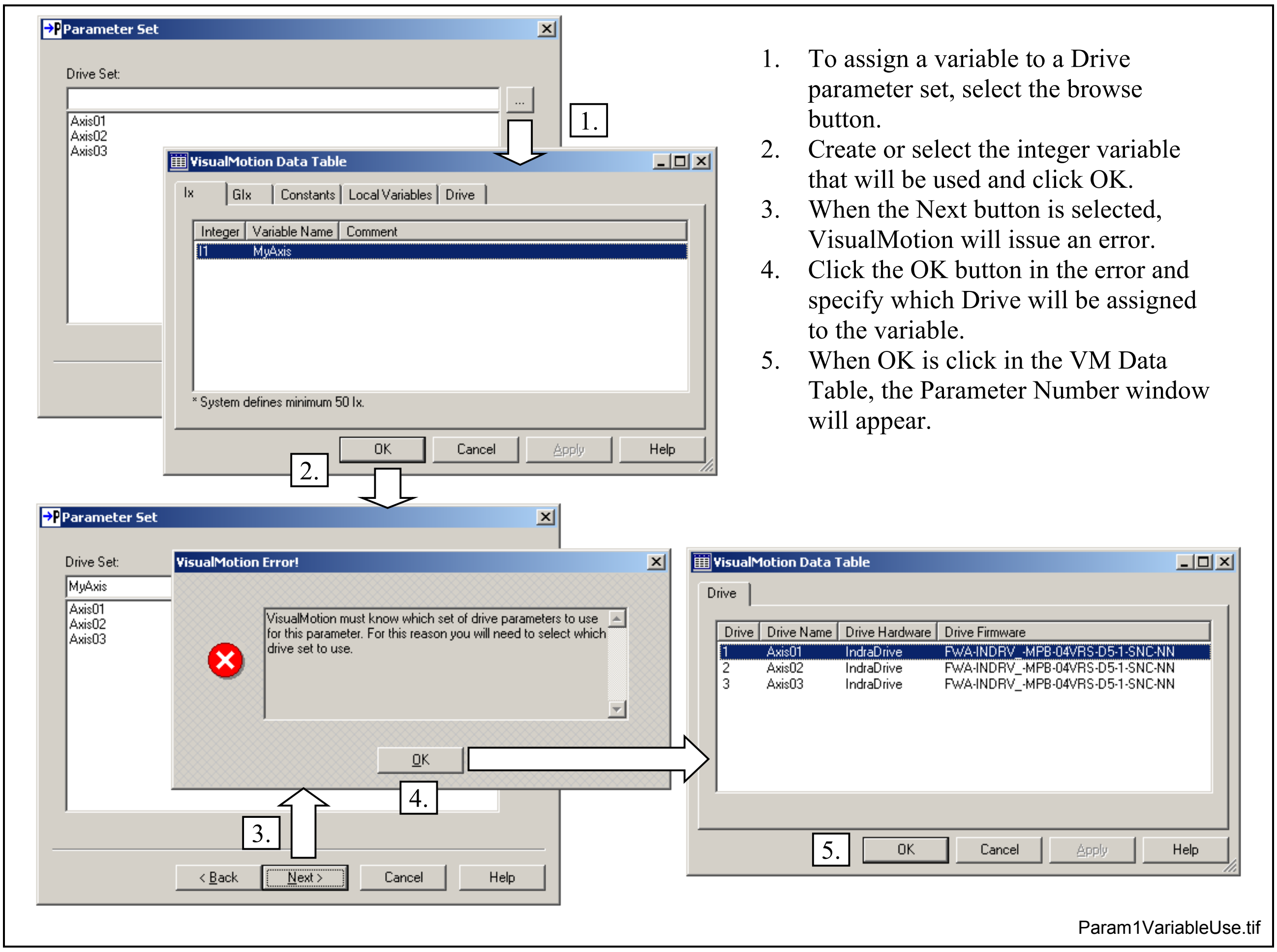


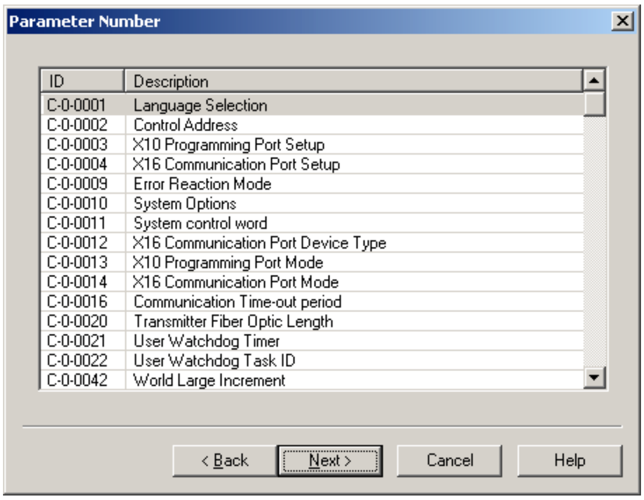
Fig. 14-48: Assigning a Variable to a Drive Parameter Set

Parameter Number

The *Parameter Number* window is used by all parameter types to select an individual parameter. When opened, the *Parameter Number* window loads a list of valid control, axis, task, or drive (S or P) parameters. Double clicking on a specific parameter opens the *Parameter Value* window.

The list of valid parameters is filtered using the following criteria:

- The location of the icon (e.g., Initialization task, or task A-D)
- The parameter type selected (e.g., C, T, A, S or P parameters)
- The parameter transfer operation being performed (read, write, or command)
- The attributes for the supported parameter (e.g., type, r/w, phase)



Cparameter.tif

Fig. 14-49: Valid Control Parameters

Parameter Value

The *Parameter Value* window is used to enter the desired value for the selected parameter. The information displayed in the Parameter value window depends on the parameter type and operation being performed. Refer to the following table for details:

Operation	Valid Data Types
Transfer Data Into a Parameter	integer or float number and labels
Transfer Data From a Parameter to a Variable	must be a variable (Fx, Ix, GIx, GFx, …)

Table 14-12: Valid Data Types

The following figure shows some examples of the different windows that can be displayed:

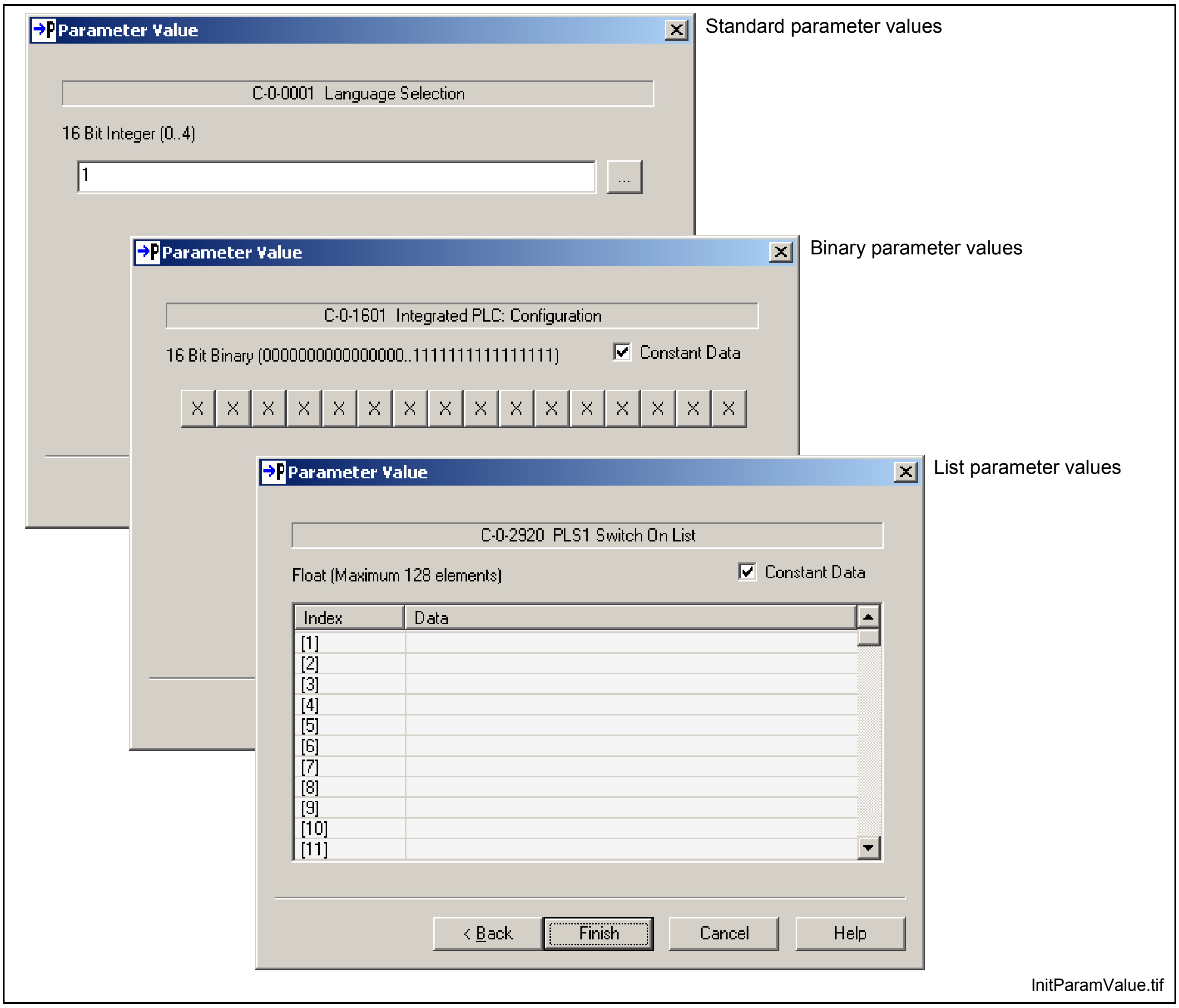


Fig. 14-50: Parameter Values

Transfer a Standard Numeric Parameter

Standard parameters are defined as parameters with a single integer or float numeric value. The value of these parameters can be transferred between a parameter and a user-defined variable.

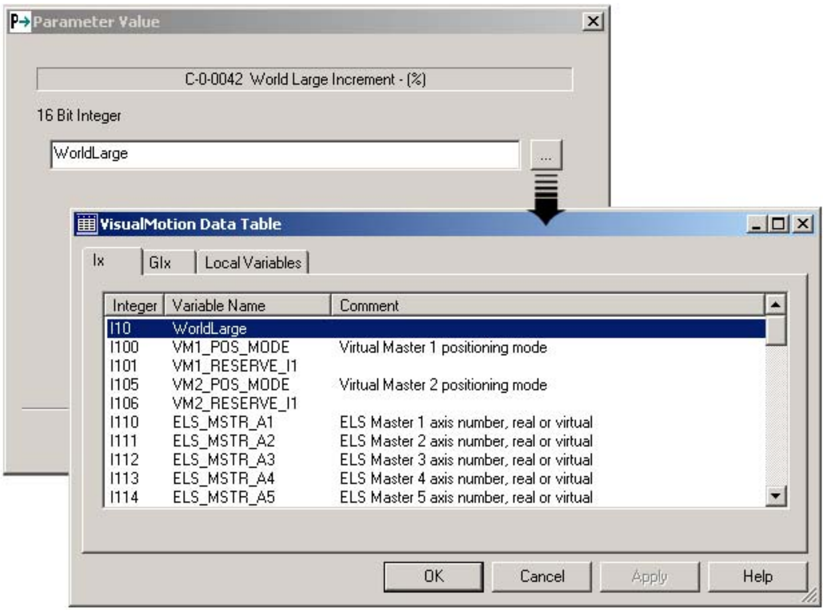
Transfer a Standard Parameter to a Variable

The following steps provide an overview of how to setup the Param1 icon when transferring a parameter value to a variable:

1. Select and place the **Param1** icon into the relevant task, subroutine, or event function.
2. From the *Parameter Transfer* window, right click on any blank row and select **Append**.
3. From the *Parameter Type* window, select the relevant parameter type (Control or Drive) and select the **Transfer Data From a Parameter into a Variable** radio button.

Note: If selecting an axis, task, or drive parameter, select the parameter set to be accessed. Refer to Parameter Type and Parameter Set on page 14-50 for details.

4. From the *Parameter Number* window, double click on the parameter number that will be read when the Param1 icon is executed.
5. Enter the name of a valid variable label (e.g., I10 or WorldLarge) or click on the browse ... button to open the VM Data Table and create or select the label from the list.



TransferParamToVar.tif

Fig. 14-51: Example: Transfer C-0-0042 to Variable "WorldLarge"

Transfer a Variable to a Standard Parameter

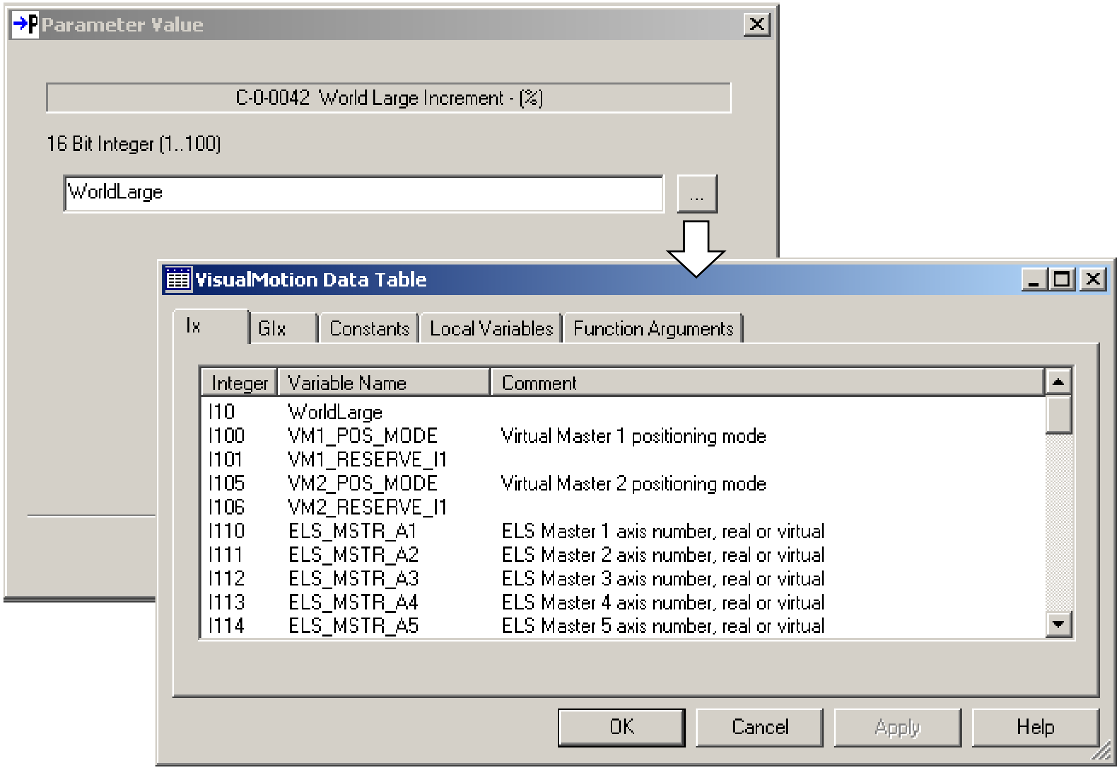
The following steps provide an overview of how to setup the Param1 icon when transferring a variable value to a parameter:

6. Select and place the **Param1** icon into the relevant task, subroutine, or event function.
7. From the *Parameter Transfer* window, right click on any blank row and select **Append**.

8. From the *Parameter Type* window, select the relevant parameter type (Control or Drive) and select the **Transfer Data into a Parameter** radio button.

Note: If selecting an axis, task, or drive parameter, select the parameter set to be accessed. Refer to Parameter Type and Parameter Set on page 14-50 for details.

9. From the *Parameter Number* window, double click on the parameter number that will be written to when the Param1 icon is executed.
10. From the *Parameter Value* window, enter the name of a valid variable label (e.g., I10 or WorldLarge) or click on the browse ... button to open the VM Data Table and create or select the label from the list.



TransferVarToParam.tif

Fig. 14-52: Example: Transfer Variable "WorldLarge" to C-0-0042

Transfer an ASCII String to a Text Parameter

The Param1 icon supports the transfer of an ASCII string to a control text parameter. Control text list parameters are **not** supported. ACSII strings cannot be transferred from a text parameter into a variable. The following control text parameters are supported:

Control Parameter	Description
C-0-0126	Date and Time
C-0-0142	Control Name NOTE:
C-0-0400	Ethernet 1 IP Address
C-0-0401	Ethernet 1 Subnet Mask
C-0-0402	Ethernet 1 Gateway IP Address
C-0-0403	Ethernet 1 Duplex Mode
C-0-0411	Ethernet 2 IP Address
C-0-0412	Ethernet 2 Subnet Mask
C-0-0413	Ethernet 2 Gateway IP Address
C-0-0416	Ethernet 2 Duplex Mode

Table 14-13: Read/Write Control Text Parameters

Note: The length of any single string in control parameter C-0-0142 must not exceed 20 characters. If the desired name has a word longer than 20 characters, break it up with a space.

Transfer an ASCII String to a Control Text Parameter

The following steps provide an overview of how to setup the Param1 icon when transferring an ASCII string to a control text parameter:

1. Select and place the **Param1** icon into the relevant task, subroutine, or event function.
2. From the *Parameter Transfer* window, right click on any blank row and select **Append**.
3. From the *Parameter Type* window, select the relevant control parameter type (C – System) and select the **Transfer Data into a Parameter** radio button.
4. From the *Parameter Number* window, double click on a valid control text parameter number that will be written to when the Param1 icon is executed.
5. From the *Parameter Value* window, enter the ASCII string and click on the **Finish** button.

Note: The limit of characters that can be entered is dependent of the selected control text parameter.

Transfer a Binary Parameter

The Param1 icon supports the transfer of 16-bit and 32-bit binary parameters. Most binary parameters in the VisualMotion system are 16-bit. Some product-specific (P) drive parameters can be 32-bit in size.

Transfer a Binary Parameter to a Variable

The steps for transferring a binary parameter to a variable are the same to those of a standard parameter. Refer to Transfer a Standard Numeric Parameter on page 14-55 for details.

Transfer a Variable to a Binary Parameter

The following steps provide an overview of how to setup the Param1 icon when transferring a variable value to a binary parameter:

1. Select and place the **Param1** icon into the relevant task, subroutine, or event function.
2. From the *Parameter Transfer* window, right click on any blank row and select **Append**.
3. From the *Parameter Type* window, select the relevant parameter type (Control or Drive) and select the **Transfer Data into a Parameter** radio button.

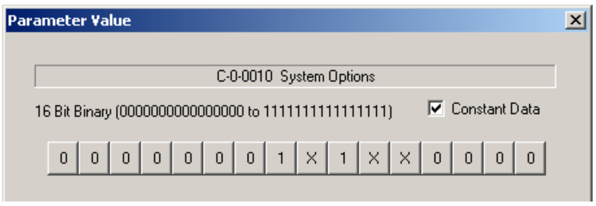
Note: If selecting an axis, task, or drive parameter, select the parameter set to be accessed. Refer to Parameter Type and Parameter Set on page 14-50 for details.

4. From the *Parameter Number* window, double click on the parameter number that will be written to when the Param1 icon is executed.
5. After selecting a binary parameter from the *Parameter Number* window, set the state of each applicable bit to the desired value in the *Parameter Value* window.

By default the **Constant Data** checkbox is selected. It allows the user to enter a constant value for a binary parameter.

The allowable settings for each bit are as follows:

- 0 (off state)
- 1 (on state)
- X (unchanged)



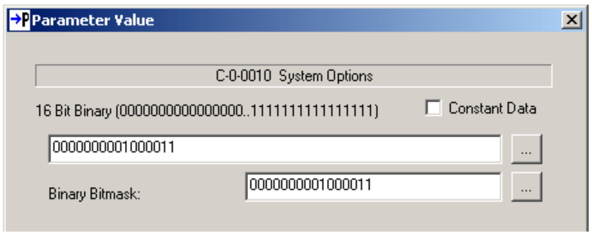
ParamConstData.tif

Fig. 14-53: Selecting Constant Data for a Binary Parameter

Note: The constant data checkbox only appears when transferring data into a parameter.

Unchecking constant data allows the user to enter a variable label from which the value will be obtained.

For binary parameters, the user can specify the state of each bit. The **Binary Bitmask** for each bit must be set to 1 in order for the bit to be processed.



ParamNotConstData.tif

Fig. 14-54: Selecting Non-Constant Data for a Binary Parameter

6. Once the binary parameter bits are set, click on the Finish button to complete the transfer setup and return to the *Parameter Transfer* window.

Transfer a List Parameter

The Param1 icon supports the transfer of integer array and float array list parameters. These list parameters only contain elements with numeric values. List parameters containing text descriptions of other parameters are not supported.

WARNING

Unexpected system errors from accessing more than one list parameter at the same time!

⇒ The accessing (reading/writing) of more than one list parameter at the same time by means of any process in the system can cause unwanted system errors which can interrupt the execution of the running program.

Transfer a List Parameter to a Variable Array

When transferring a list parameter to a variable array, the Param1 icon first reads the contents of the list parameter and then writes the elements that contain valid data to the variable array specified by the user.

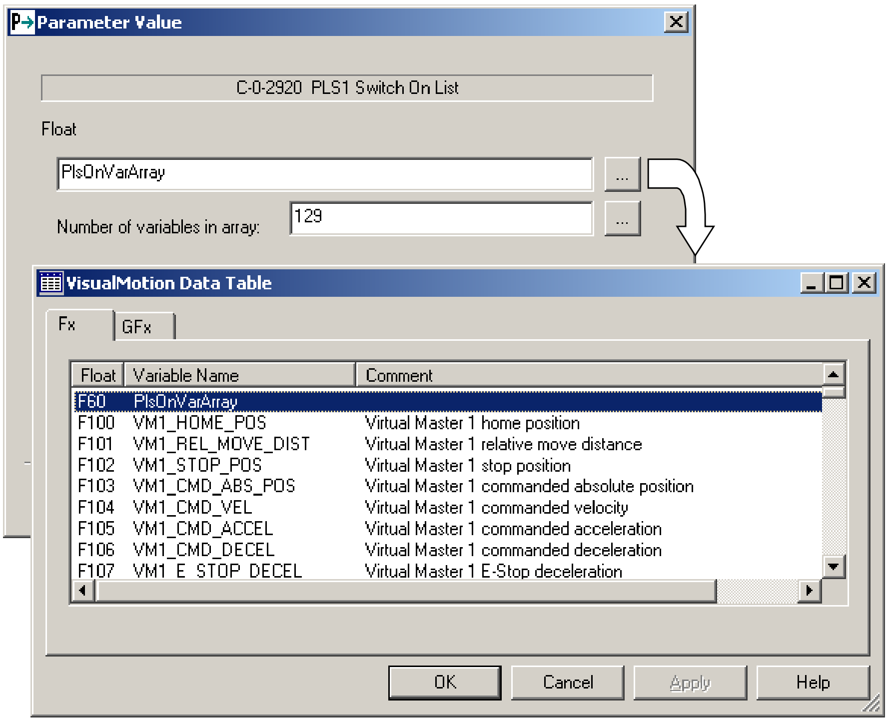
The number of elements in a list parameter varies from parameter to parameter. The user should know the exact number of elements containing valid data that will be transferred from the list parameter to the variable array. This information will help determined how many variables will be written when the Param1 icon is executed.

The size of the variable array is dependent on the number of elements that contain valid data. The user simply specifies the start of the variable array and the Param1 icon will use the necessary number of variables in the array up to the maximum number of elements in the list parameter. For this reason, allow the maximum number of elements in the list parameter to be used in the variable array.

The initial steps for transferring a list parameter to a variable array are the same as those for transferring a single parameter to a variable describe on page 14-55. The following sections will focus mainly on the functionality of the variable array once the data is written as well as some programming techniques for accesses the data from the array.

1. After the initial steps are completed, from the *Parameter Value* window, select the browse ... button to open the VM Data Table

and select or create the variable label that will be use as the starting point for the variable array.



TransParListToArr.tif

Fig. 14-55: Selecting a Variable Label from the VM Data Table

In the figure above, list parameter C-0-2920 contains a maximum of 128 elements that are available to store data. The number 129 in the "**Number of variables in array**" field represents the 128 elements that are available in the list plus a count variable that identifies how many elements actually contain valid data.

A variable array consists of a block of variables, starting at the variable (e.g., PlsOnVarArray) entered in the figure above and continuing in an ascending order up to the number appearing in the "**Number of variables in array**" field.

2. Once the variable label is selected, click the **Finish** button on the *Parameter Value* window to complete the parameter transfer setup.

Writing to a Variable Array

When the Param1 icon is executed in the program flow, the contents of the list parameter is read and written to the variable array. The information written to the variable array depends on the value that is written to the first variable in the array. This first variable is known as the count variable. Variable values can be viewed by selecting ***Data ⇒ Variables…***

The value that is written to the count variable must be done before the Param1 icon is executed. This can be accomplished by placing a Calc3 icon before the Param1 icon and writing a value directly to the count variable. If the number of elements in a list parameter, containing valid data, is known, then write this value into the count variable before executing the Param1 icon. After the parameter transfer is executed, the variable array will display the count variable first followed by the values for each element in the list parameter.

Note: If a number greater than the actual number of elements containing data is written to the count variable, the number will be overwritten with the correct value followed by the values of each element in the list parameter.

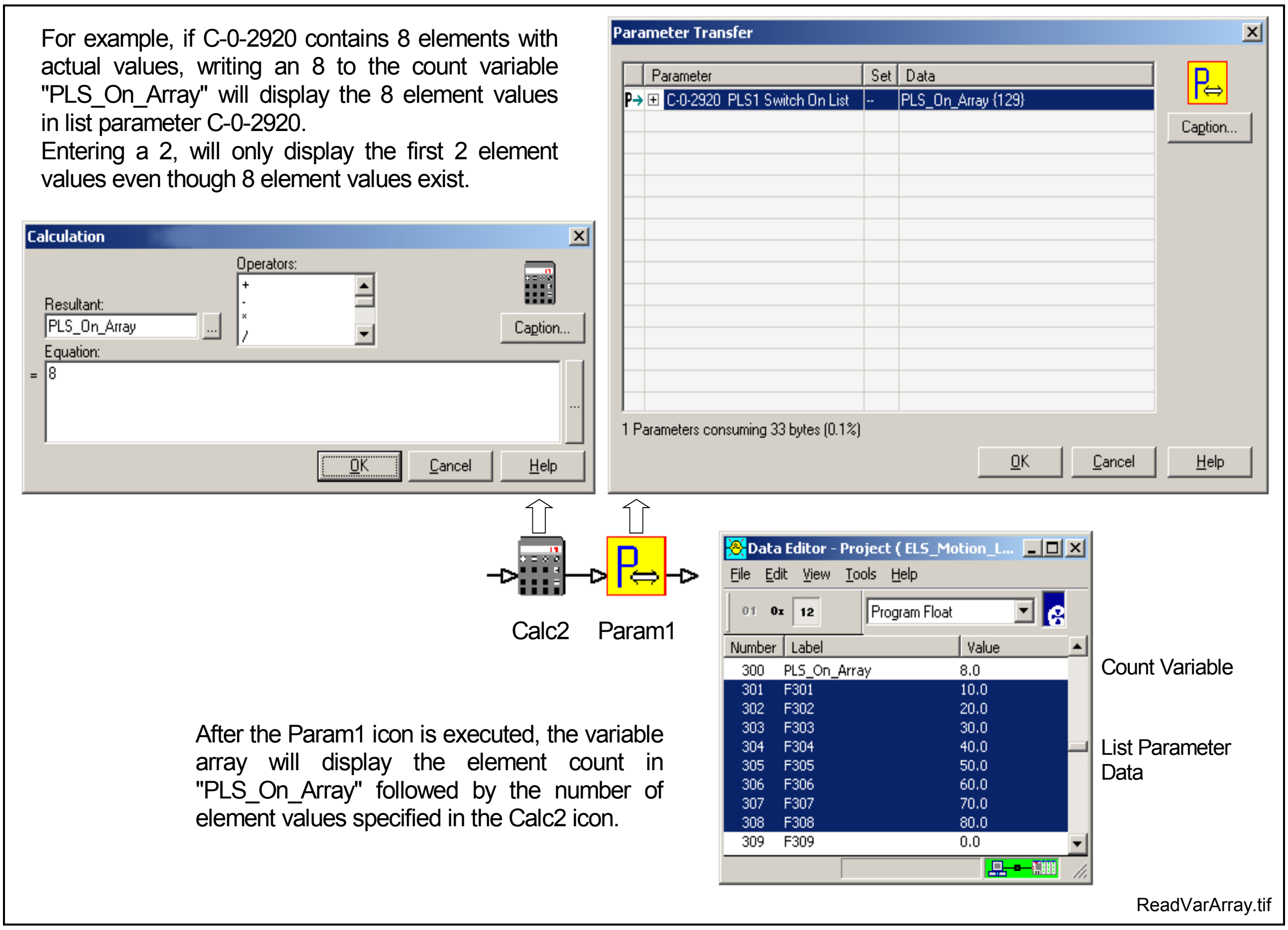


Fig. 14-56: Writing a List Parameter to Variable Array

Requesting the Size of a List Parameter

The Calc3 icon can be used together with the Param1 icon to request the number of elements containing valid data in a list parameter as well as the maximum number of elements available in the list parameter.

To accomplish this,

1. Place a Param1 icon into the relevant task, subroutine, or event function.
2. Configure the Param1 icon to **Transfer Data from a Parameter into a Variable**.
3. From the *Parameter Value* window, select the browse ... button to open the VM Data Table and select or create the variable label that will be use as the starting point for the variable array.
4. Then, place a Calc3 icon and write a 0 to the count variable selected in step 3.

These steps are illustrated in Fig. 14-58.

Note: The Param1 icon must be configured with the relevant list parameter(s).

When the Param1 icon is executed, the variable array will display a 0 in the count variable, followed by the number of elements containing valid data, and then the maximum number of elements available for the configured list parameter. Refer to the figure below for an example of the data displayed in the variable array:

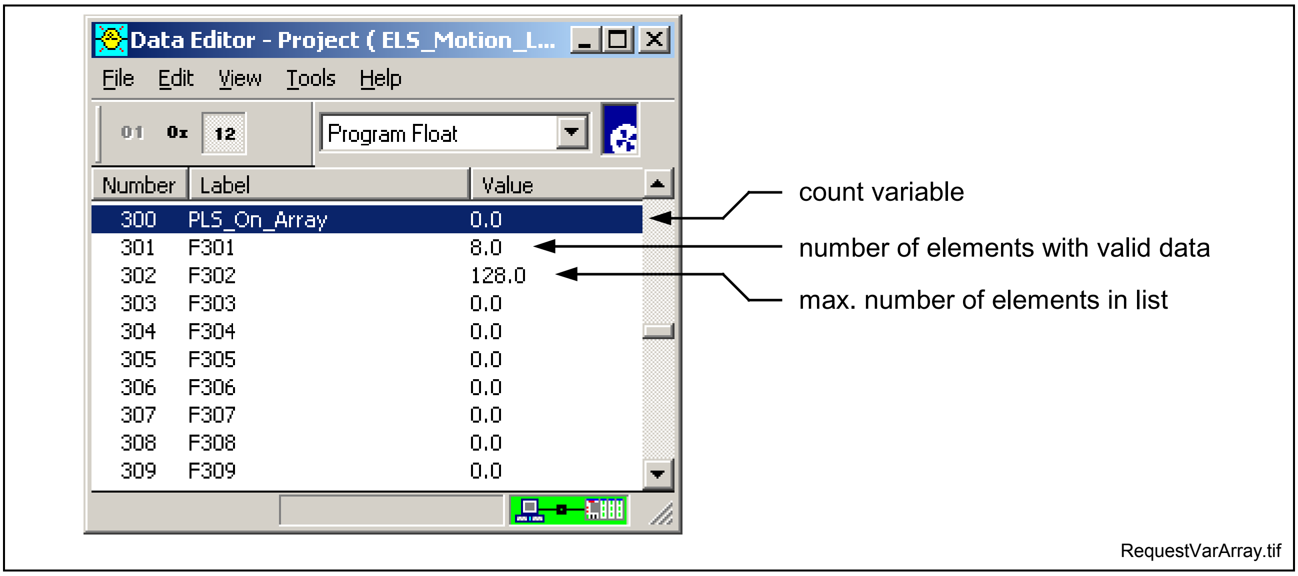


Fig. 14-57: Request Data from List Parameter

Automating the Request of a List Parameter

To automate the entire request of a list parameter simply create an icon flow that contains a pair of icons (Calc3 & Param1) requesting size data followed by a second request of valid elements to a variable array.

The key to the process is to first request the number of elements in the list parameter, as in Fig. 14-57, by writing a 0 to the count variable. The next step requires taking the value of the second variable (e.g., F301), and copying it back to the count variable before executing the second Param1 icon. Refer to the following figure for details:

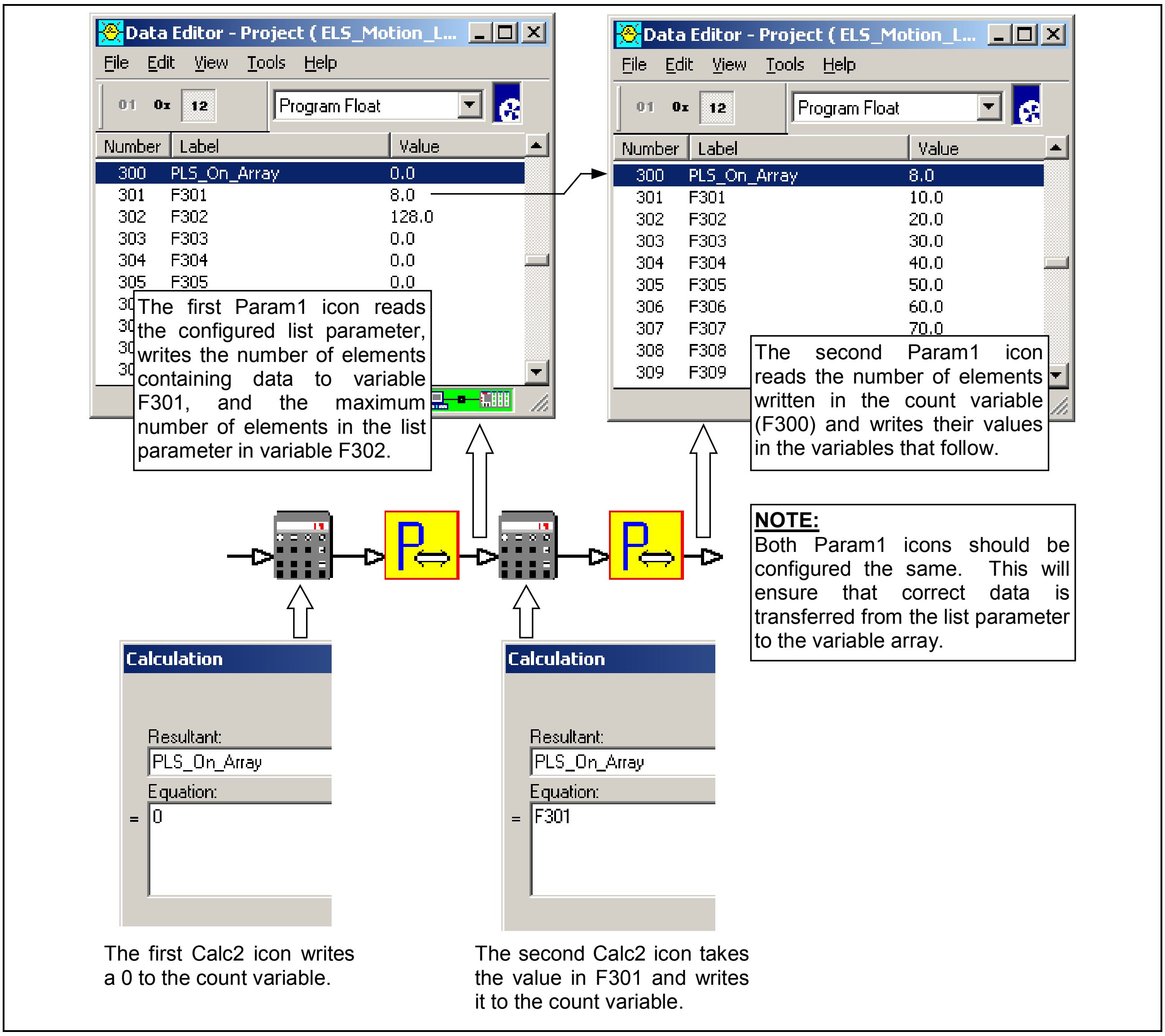


Fig. 14-58: Requesting a List Parameter to an Array

Transfer a Variable Array to a List Parameter

When transferring a variable array to a list parameter, the user must define exactly how many variables to transfer to the list parameter. The count variable in the variable array determines the number of variables that will be transferred into the list parameter when the Param1 icon is executed.

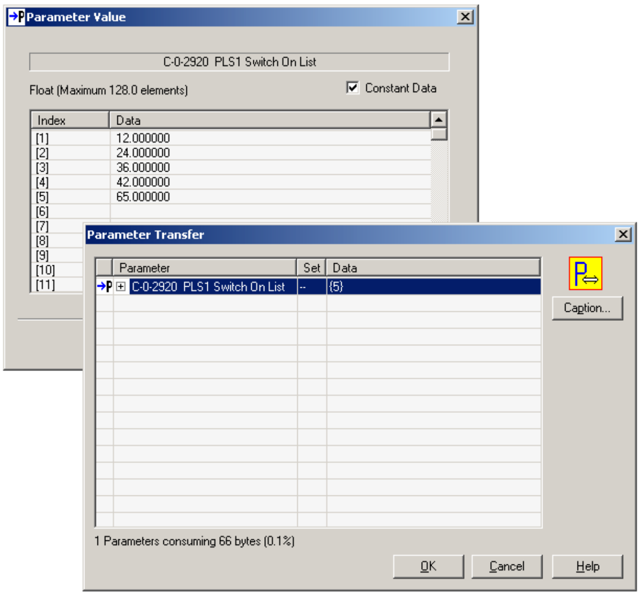
Note: Transferring a specific number of elements to a list parameter will truncate the list parameter by that number. If a list parameter contains 16 elements with data and a Param1 icon transfers 4 elements from a variable array, then the list parameter will only have 4 elements.

To transfer a variable array to a list parameter, use the following steps:

1. Place a Calc3 icon and write the number of variables to transfer in the count variable.
2. Place a Param1 icon after the Calc3 icon, select the parameter type, and select the **Transfer Data Into a Parameter** operation.
3. After selecting the list parameter, uncheck the *Constant Data* checkbox from the *Parameter Value* window. Refer to Fig. 14-60 for details.

Constant Data

For list parameters, the maximum number of elements will be displayed for the supported data type. The user can hardcode a constant value for each element by clicking on the *Data* field and entering the desired value. Once the Param1 icon is executed, only the elements containing values are written to the list parameter.



ListContData.tif

Fig. 14-59: Selecting Constant Data for a List Parameter

Constant Data Unchecked

For list parameters, only a variable label (e.g., F60 or StartVarArray) can be used. Variable labels can be entered manually, if known, selected from an existing list or created by clicking on the browse ... button. A numeric value is not allowed. This variable represents the start of the variable array.

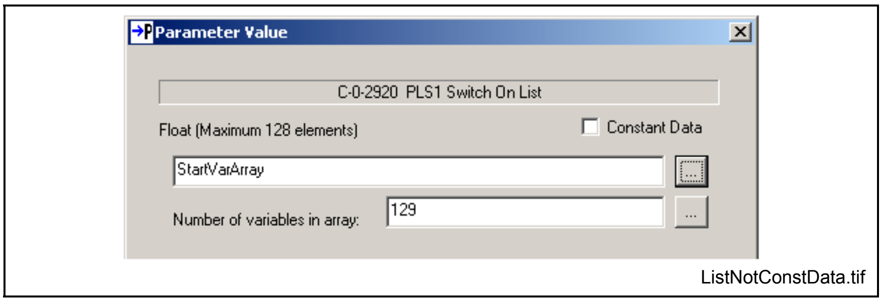


Fig. 14-60: Selecting Non-Constant Data for a List Parameter

1. Click the first browse ... button and select the variable that identifies the start of the variable array. This is the same count variable mentioned in step 1.

The figure below describes the configuration and execution listed in the above steps.

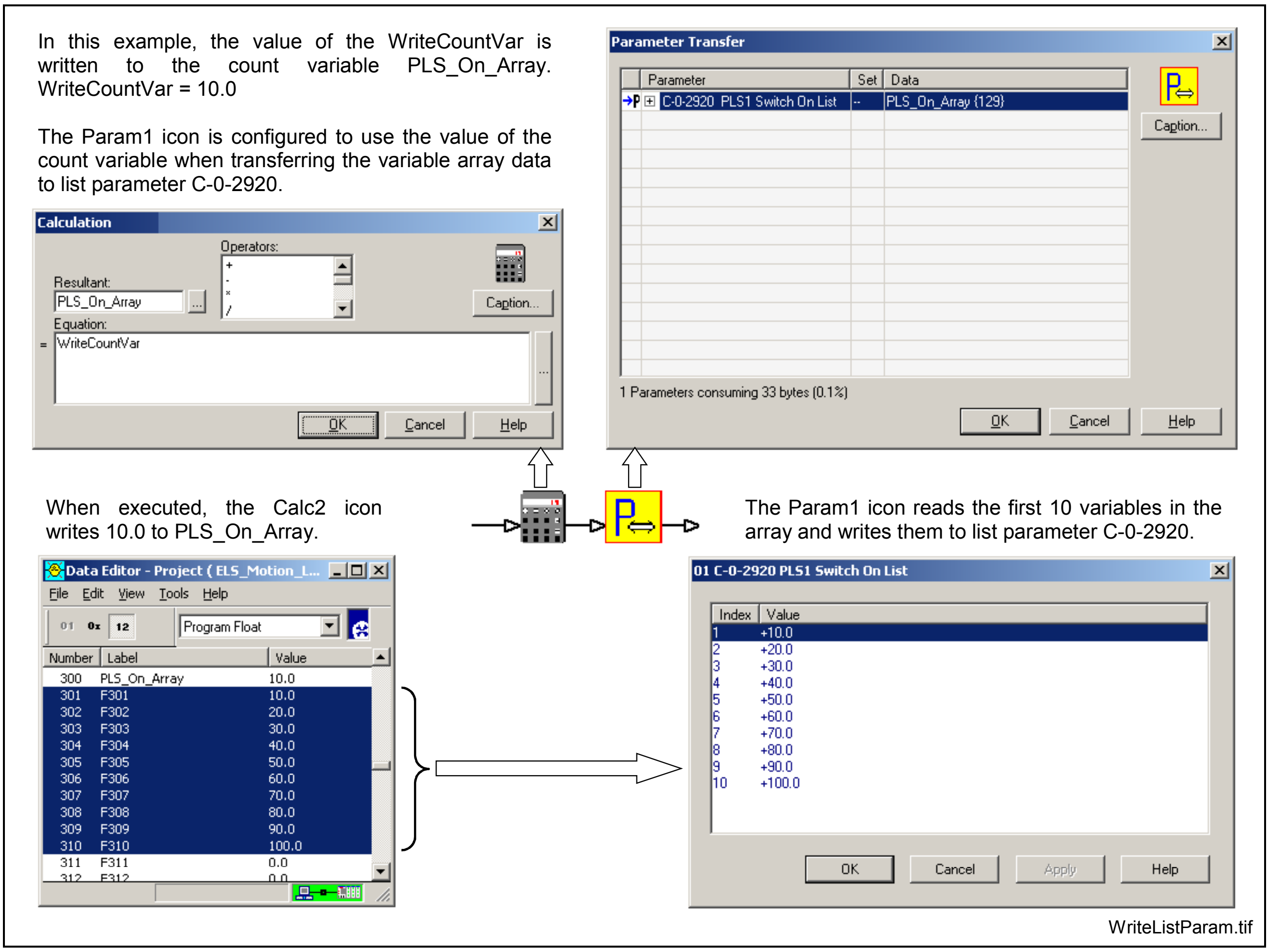


Fig. 14-61: Writing a Variable Array to a List Parameter

Rexroth Bosch Group

Execute and Reset a Command Parameter

The Param1 icon supports the execution and resetting of control and drive-resident commands.

The following steps provide an overview of how to setup the Param1 icon when executing or resetting a command parameter:

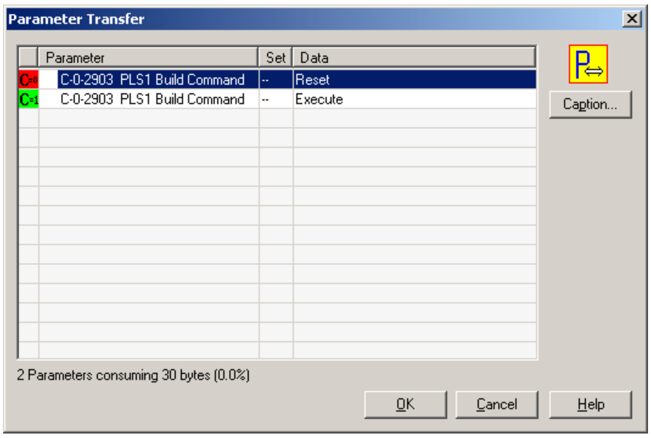
1. Select and place the **Param1** icon into the relevant task, subroutine, or event function.
2. From the *Parameter Transfer* window, right click on any blank row and select **Append**.
3. From the *Parameter Type* window, select the relevant parameter type (Control or Drive) and select the **Execute or Reset a Command Parameter** radio button.

Note: If selecting a Drive (S or P) parameter, select the drive set name that matches the relevant axis from the *Parameter Set* window. Only axes that are configured under ***Setup ⇒ Axis*** will be available for selection.

4. From the *Parameter Number* window, double click on the relevant command parameter number.
5. From the *Parameter Value* window, select the **Execute Command** or **Reset Command** radio button.

Note: When programming command parameters, it is recommended that the command parameter first be reset and then executed within the same Param1 icon. This procedure ensures that a previous execution of the command is not currently active. Command parameters are edge sensitive and require a 0 -> 1 transition to process.

Refer to the following figure for a simple example of the above mentioned procedure:



CommandParam.tif

Fig. 14-62: Command Parameter Reset and Execute Example

The list of valid command parameters is filtered using the following criteria:

- Where the Param1 icon is placed (e.g., Initialization task, or task A)
- The parameter type selected (e.g., C, S or P parameters)

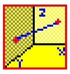
The following table lists the supported control command parameters and whether or not they can be used as part of an Initialization or Runtime program.

Control Parameter	Description	Initialization (Phase 2)	Runtime (Phase 4)
C-0-0082	Save global variables command	**X**	
C-0-0167	Save built CAM to flash command	**X**	**X**
C-0-0170	NVRam backup command	**X**	
C-0-0172	Compact flash backup command	**X**	
C-0-0174	Compact flash restore command	**X**	
C-0-0810	TPT message and prompt control word	**X**	**X**
C-0-0814	TPT data transaction word	**X**	**X**
C-0-0993	Software reset for PPC	**X**	
C-0-0994	Shutdown command for flash programming	**X**	
C-0-0996	Clear program and data memory	**X**	
C-0-0997	Clear diagnostic log	**X**	
C-0-2522	Oscilloscope trigger control word	**X**	**X**
C-0-2903	PLS1 build command	**X**	**X**
C-0-2905	PLS1 activate command	**X**	**X**

Table 14-14: Control-resident Command Parameters

Note: Drive-resident command parameters vary based on drive type (e.g., Rexroth Diax 04, EcoDrive 03, IndraDrive) and firmware selected when the axis was configured under ***Setup*** ⇒ ***Axis*** in the Project Navigator.

Path



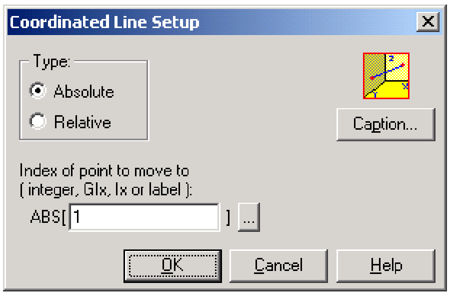
The Path icon is used to set up multi-axis coordinated straight line motion. Placing a Path icon on a task or subroutine workspace automatically displays a *Coordinated Line Setup* window.

Motion may be Absolute or Relative and is defined by the two endpoints of the line of motion. Points are specified by an index into the point table using an integer constant, variable, global variable or an equivalent label. Refer to the Calc3 icon. The second point, or target point, must be a point on the point table. The first point is the current position, which can be anywhere.

An absolute move begins from the endpoint of the previous path segment, or current position if the system is halted, and terminates at the absolute point specified.

The relative move begins at the endpoint of the previous path segment, or current position if the system is halted, and terminates at the relative offset point specified.

Clicking the browse button to the right of the **ABS** or **REL** field opens the VM Data Table.
A Go1 icon is not required when using the Path icon. When executed, the motion will immediately be sent to the path planner. Stepping to the next icon will take place immediately.



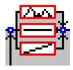
Path.tif

Fig. 14-63: Path Setup

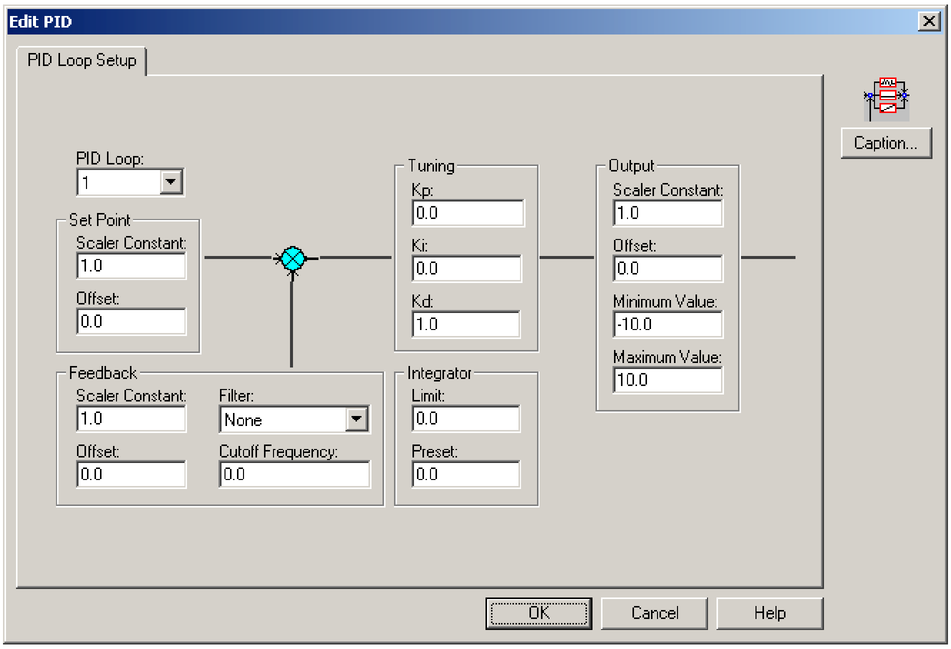
Coordinated Motion Events

Event triggers for coordinated motion are associated with the points table. For specific information on *Coordinated Motion Events,* refer to section 5.2 in volume 1 of the *VisualMotion 11 Functional Description*.

PID2



The PID2 icon can only be placed in the Initialization task and is used to modify an existing PID Loop initially configured under Setup. The program flow will be slowed if this icon is continuously processed, so it desirable to execute this icon only once per PID loop. The PID functions load on GPP resources is included in Load Due To I/O control parameter C-0-0202.



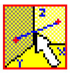
PID.tif

Fig. 14-64: PID Setup

Refer to section 4.7, *PID Loops Setup*, in volume 1 for details. For specific information on PID functionality, refer to section 5.4, *PID*, in volume 1.

Rexroth Bosch Group

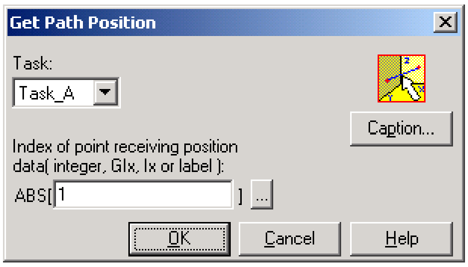
Position



The Position icon is used to obtain the current position of one of the tasks from the path planner. Once this position is read, the value is stored in an absolute point table. This is useful when operating the system in a teach mode without a Teach Pendant. It also could be used to check axis position while running a program. For instance, a loop could be setup whereby the path position is constantly obtained and compared to a target position. When the path position matches the target position, a bit is toggled. The toggle causes a new branch test result, which alters program flow.

Once this icon captures the current position into a point, Calc3 icons can be used to copy the attributes of these points to new ones. Single axis motion can be created from coordinated motion data and offset or speed change adjustments can be made automatically.

Placing a Position icon on a task or subroutine workspace automatically displays a *Get Path Position* window. Four control tasks (A-D) can be selected from the Task menu. The absolute point table destination for the position data may be specified by an integer constant, variable, global variable or an equivalent label. Clicking the browse button to the right of the **Index of point** field opens the VM Data Table.



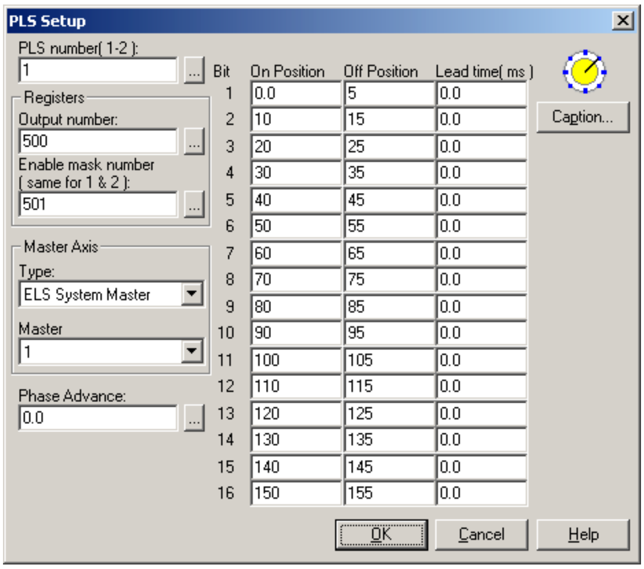
Position.tif

Fig. 14-65: Get Path Position

PLS2

Two programmable limit switch (PLS) data structures exist in each user program with initial values of zero. The PLS icon is used to initialize these values when encountered in the Initialization task or in any runtime task. Up to 16 outputs and a phase advance are supported. The phase advance is added to all positions in the PLS table. The position input for the PLS can be an ELS Master, ELS Group, or Real Master.

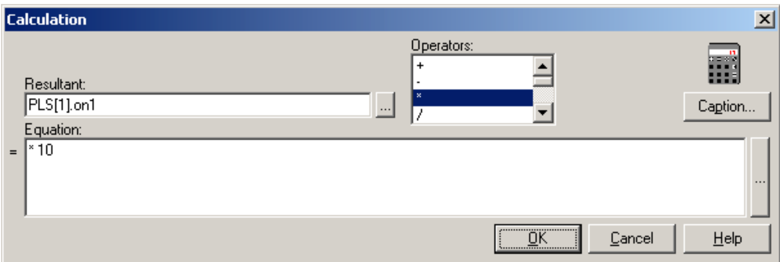
Both control PLS process are active once the program is running. The outputs are active only if a valid master is assigned, the mask register bits are set to 1, and the on/off positions are not zero or equal to each other. The outputs are updated every Sercos cycle. The outputs are disabled once the mask register bits are set to zero.



PLSicon.tif

Fig. 14-66: PLS Setup

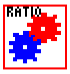
It is possible to write to the on/off positions in the PLS table from the user program using the Calc icon. Refer to PLS Data when using the Calc3 icon for details.



ControlPLSCalcUpdate.tif

Fig. 14-67: Using a Calc icon to Write to a PLS

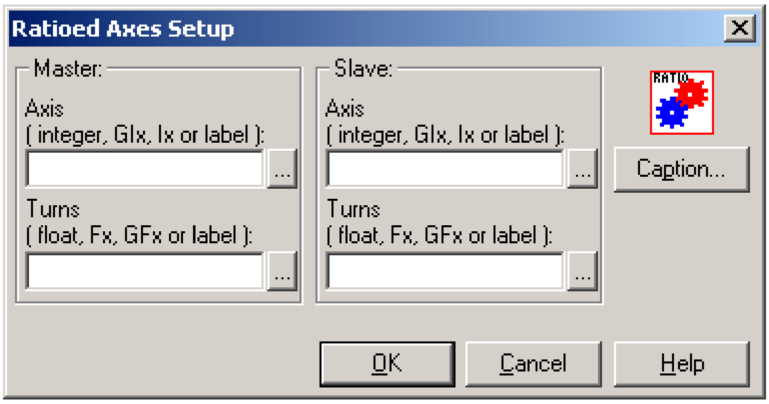
Ratio



The Ratio icon is used to set the ratio between two axes in a master/slave relationship, as when a gantry robot has a motor on each side of a supporting circular track. It can be used in Tasks which also contain coordinated axes.

The Ratio icon may also be used to link several axes to the same master axis or to chain several drives together. For example: if drive 1 is master to drive 2, drive 3 can be made a slave to drive 2, thereby linking drive 3 to drive 1 through drive 2. Note that the response time of drives chained in this manner is additive, at least one Sercos cycle (approximately 2ms) must occur between each master to slave link.

Placing a Ratio icon on a task or subroutine workspace automatically displays a *Ratioed Axis Adjust* window**.** The Master and Slave Axes should already be assigned as Ratioed axes under ***Setup ⇒ Axes***. Refer to the *Ratio Axes Setup* in section 4.3 of volume 1 for details.



Ratio.tif

Fig. 14-68: Ratio Setup

The **Master** and **Slave** axes are selected by entering an integer constant, variable, global variable or an equivalent label in the **Axis** data fields.

The VM Data Table is available for all four fields. One axis is selected as a master axis. A slaved axis must not be assigned to any task other than the task containing the master axis.

Rotation of the master axis controls the proportional rotation of the selected slave axis according to the formula:

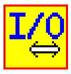
Slave axis velocity = Master axis velocity × (Slave ratio factor ÷ Master ratio factor)

The **Master** and **Slave Turns** fields permit simple entry of the ratio between the axes. The ratio factors may be a float constant, variable, global variable or an equivalent label. Individual data boxes for master and slave eliminate the need to normalize the ratio. For example, simply entering the number of teeth on each of two meshed gears allows VisualMotion to calculate the necessary coefficient. Each factor is in float format and is normalized before the division operation. This insures that the calculation maintains maximum precision with repeating decimals such as 2/3.

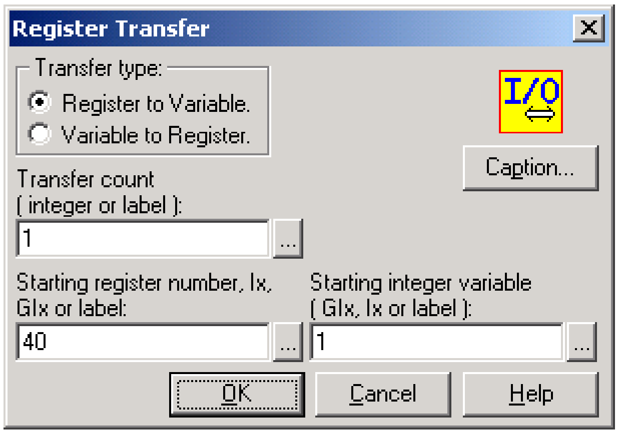
Entering the axis ratio factors using the Ratio icon automatically updates the master factor parameter A-0-0031 and slave factor parameter A-0-0032.

By default, the slave axis is maintained in the drive's position loop mode at all times, even when the user program is not running. The drive's shaft position remains locked to the master axis within the torque limits of the drive.

Reg (Register Transfer)



The Reg. icon is used to transfer data between the I/O registers and either the integer variable or global integer variable table. One to 1024 registers (16-bits each) may be transferred with a single command. Placing a Reg. icon on a task or subroutine workspace automatically displays a *Register Transfer* window.



Reg_Transfer.tif

Fig. 14-69: Register Transfer

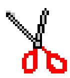
Example

This transfers a 15 bit DEA card input register to a variable.

Transfer type selects the direction of data transfer between the I/O register(s) and a control integer variable table. Transfer count specifies the number of consecutive 16-bit words to be transferred. **Starting register number** specifies the base, or lowest address of the source for the start of the transfer. **Starting integer variable** specifies the base address of the target of the transfer.

The **Transfer count**, Starting register number and starting integer variable must be an integer constant, variable (Ix), global variable (GIx), or an equivalent label.

Scissors

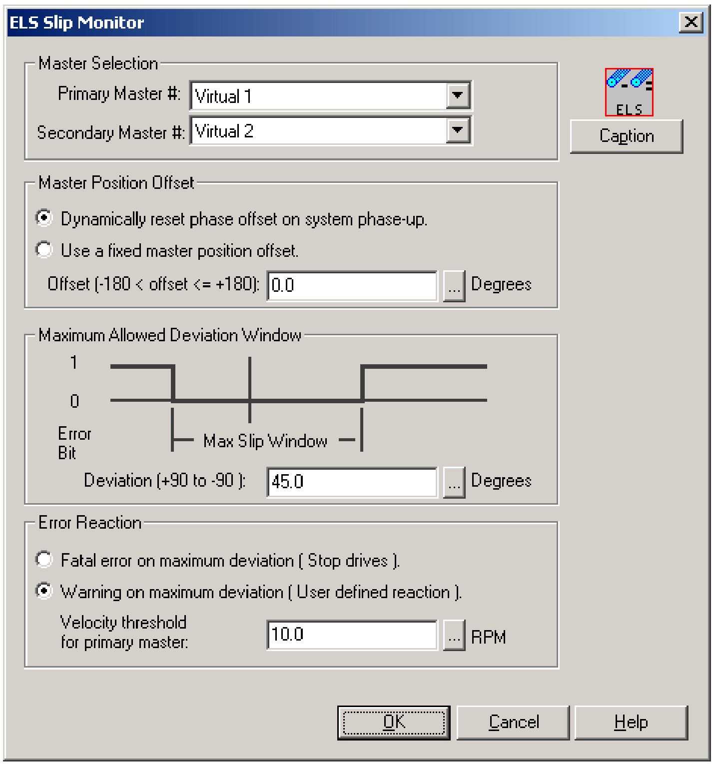


The Scissors icon is used to delete a line between two icons. Select the Scissors on the palette, position the tip of the Scissors cursor over the line to delete, press the left mouse button to delete.

Slip Monitor

The Slip Monitor Setup feature is used to monitor the difference in position (phase) between two ELS System Masters and initiates an error reaction when the difference is outside the allowable deviation window.

The ELS System Master Slip Monitor feature is initially configured under ***Setup ⇒ Processes ⇒ ELS ⇒ System Masters ⇒ Slip Monitor***. The icon is used to modify an existing configuration in the Initialization task. When executed in the Initialization task program flow, any initialization data configured under setup will be overwritten.



SlipMonitorSetup.tif

Fig. 14-70: Slip Monitor Setup

Refer to *Slip Monitoring for ELS System Master* in section 6.1 of volume 1 for details.

Start1

All four control tasks A-D, initialization task, subroutines and event functions must begin with a Start1 icon. The Start1 icon indicates the beginning of program flow to the control compiler and can also be used to declare function arguments and local variables.

Note: The ***Define Function Arguments…*** button is only available for subroutines (initialization and standard).

The ***Properties…*** button is only available for Task A-D.

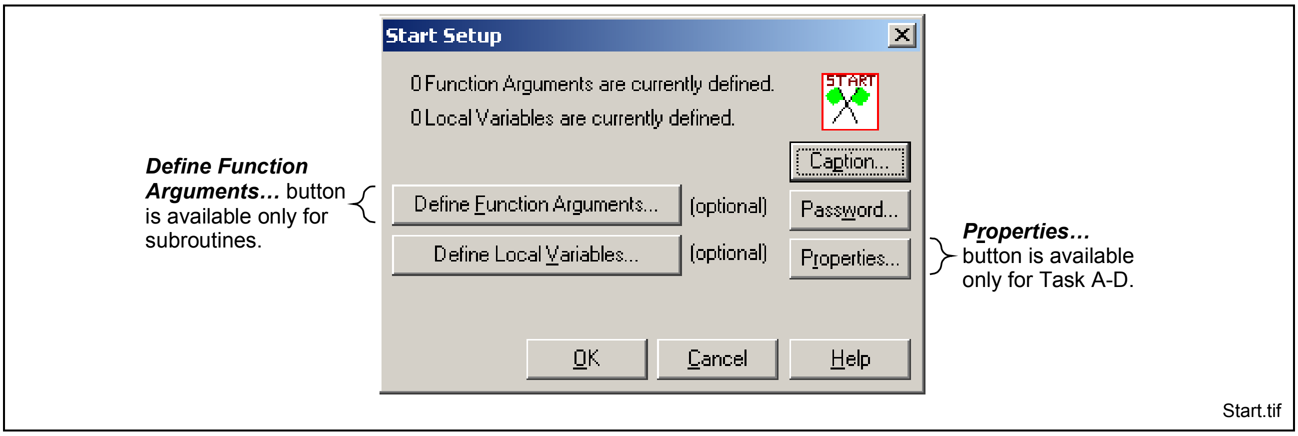


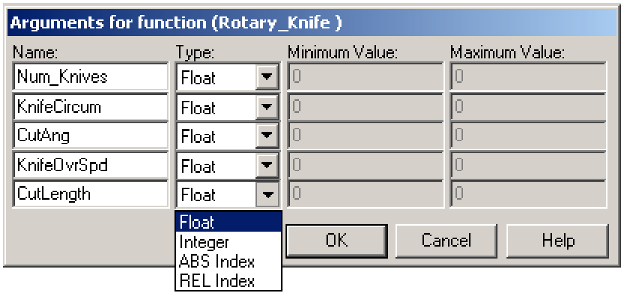
Fig. 14-71: Start Setup

Define Function Arguments

Function arguments are stack-based variables whose values exist only while in the subroutine (initialization or standard) where they are declared. The values of stack-based variables are placed in a memory location shared by function arguments and local variables. The total number of combined function arguments and local variables in a subroutine is limited to 16.

Selecting the ***Define Function Argument…*** button allows the user to define up to 5 function arguments of type **Float**, **Integer**, **ABS Index** and **REL Index**. Remember that the only thing that is being defined is the **Name** and **Type** for the function argument. The value that is passed to the defined function argument comes from the Sub1 icon of the calling task or subroutine.

Note: Duplicate function argument labels are not allowed within the same subroutine. A compiler error will be issued when the program is compiled. It is also recommended that the programmer not use VisualMotion keywords when naming function arguments.



FunctionArgu.tif

Fig. 14-72: Function Arguments

Function Argument Types

The available data types for function arguments are Float, Integer, ABS Index and REL Index. When **Float** and **Integer** are defined, the value passed by the caller (using the Sub1 icon) is lost when the subroutine ends.

Note: VisualMotion allows a single function argument value to be returned to the calling task or subroutine when using the Optional Return Value feature in the Sub1 and Finish icons. Refer to Optional Function Arguments (Sub1) on page 14-81 for details.

When **ABS Index** or **REL Index** are defined, the value that is passed is the index number of the absolute or relative points table and not an actual point value. The index number can be used within the subroutine and will not be saved when the subroutine ends. However, if any point data is modified, using an Calc icon, the value for that point will be modified in the project, even after the subroutine ends.

Define Local Variables

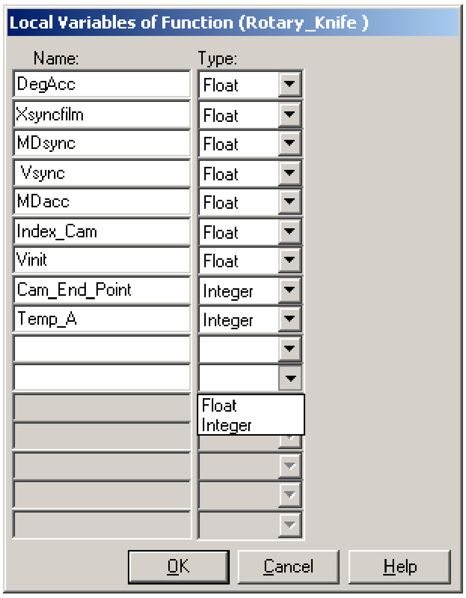
Local variables are stack-based variables whose values exist only while in the subroutine (initialization or standard) and/or event function where they are declared. Local variables are used within the subroutine for local data only and do not exists outside the subroutine. Local variables are useful for temporary results within the subroutine and their values are lost when the subroutine ends.

The total number of combined local variables and function arguments in a subroutine is limited to 16. The maximum number of local variables is dependent on the number of defined function arguments. If a maximum of 5 function arguments are defined, then only 11 local variables can be defined.

Notice that in the following figure, only 11 local variables are available, the rest of grayed out. This is because 5 function arguments were defined in Fig. 14-72 for the same subroutine.

Note: Duplicate local variable labels are not allowed within the same subroutine. A compiler error will be issued when the program is compiled.

It is also recommended that the programmer not use VisualMotion keywords when naming local variables.



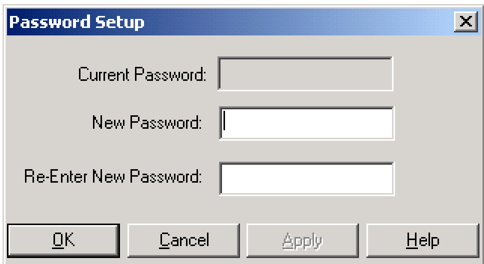
LocalVariables.tif

Fig. 14-73: Local Variables

Password

Any task, subroutine or event function can be password protected. The following *Password Setup* window is opened when the ***Password…*** button is clicked in the *Start Setup* window.

Important Note: Once a password is entered, it should be record and saved. The system does not allow entry to a program area without the correct password. If the password is forgotten for a subroutine or event function, the function must be deleted (right-click on name) and recreated. If the password is forgotten for a task (Initialization, A-D), the entire project must be recreated.



PasswordSetup.tif

Fig. 14-74: Password Setup

The password can be any combination of letters and/or numbers up to a maximum of 20 characters. Once set, the password is request the first time the task, subroutine or event function is viewed. Afterwards, the user can switch between task without having to enter the password.

Password Protection

Once the password is set, the user must enter the valid password to gain access to desired workspace. If an invalid password is entered or the ***Cancel*** button is clicked, the following message is displayed in the icon workspace. Password protected workspace areas are indicated by a password icon in the Project Navigator window.

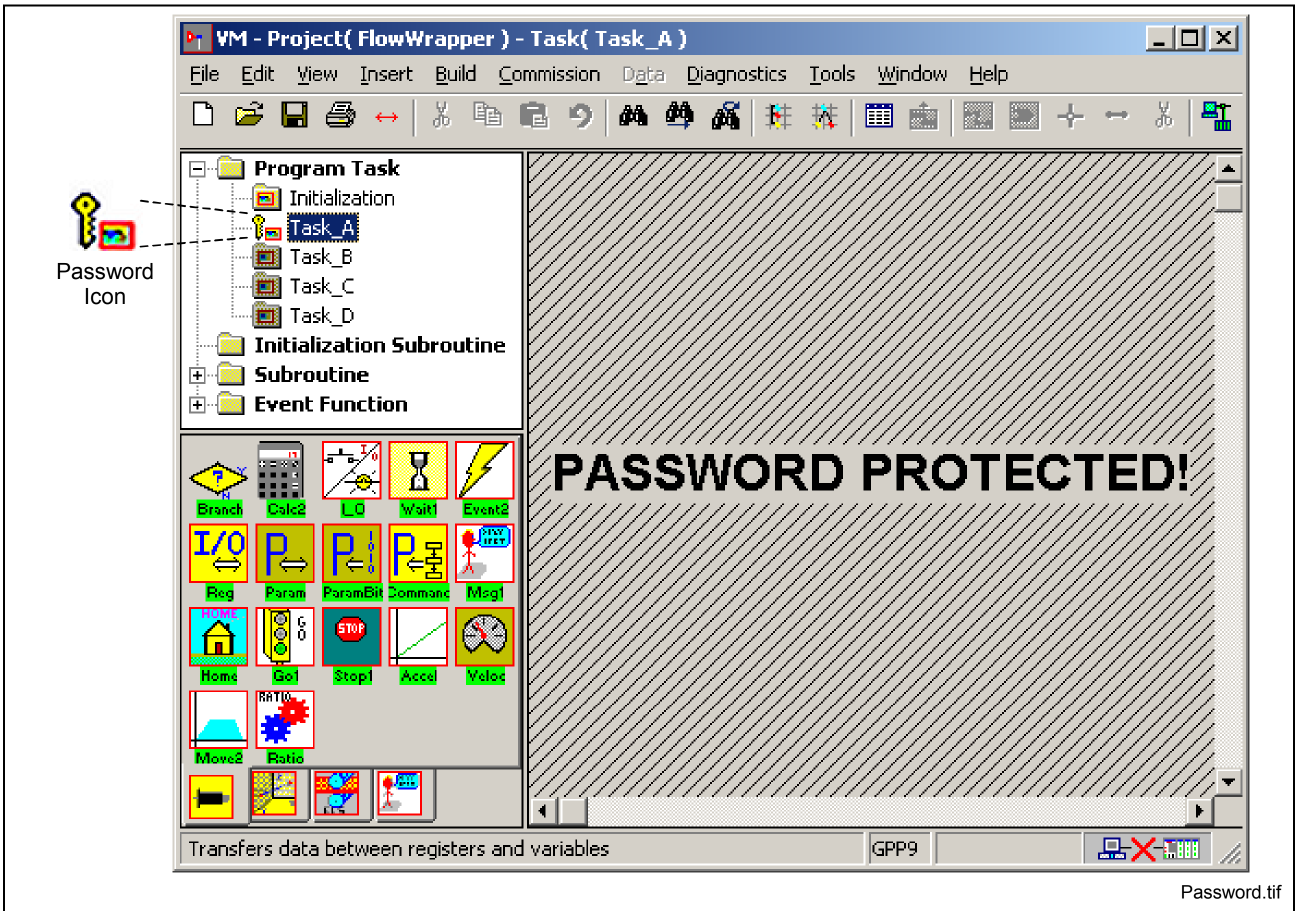


Fig. 14-75: Password Protected

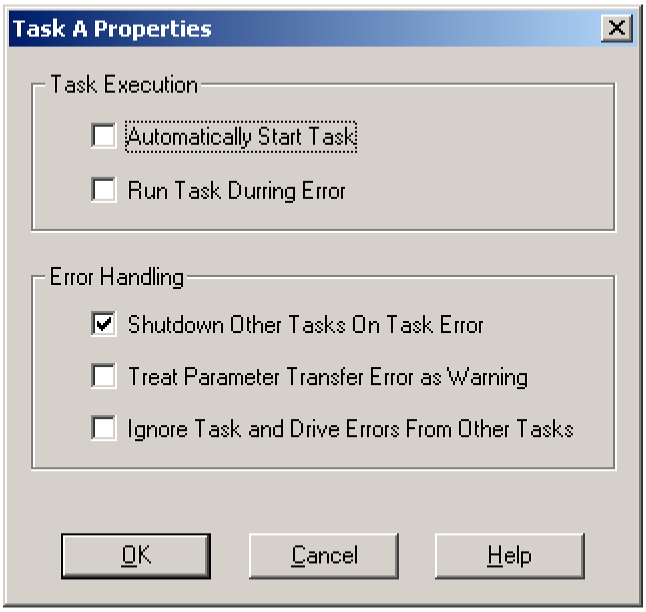
Clear Password

Open the *Password Setup* window by clicking the ***Password*** button. The **Current Password** field will show the length of the set password as asterisks. To clear the set password, simply click on the ***OK*** button. Clicking the ***Cancel*** button will not clear the password.

Properties

The ***Properties…*** button is used to set task related options for Task A-D. The **Start** icon of each task (A-D) can have its own specific task options. Selecting any checkbox, set the corresponding bit in task parameter T-0-0002 for each task (A-D). Refer to task parameter T-0-0002 for a description of each option available.

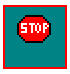
Note: The *Task Properties* window can also be open by right clicking over a task letter in the Project Navigator window.



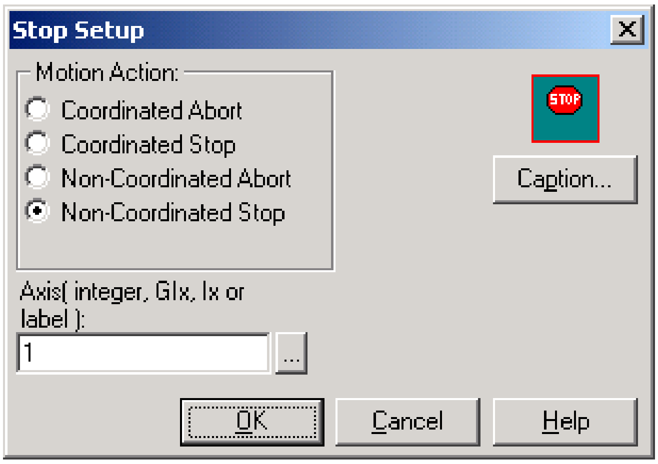
TaskProperties.tif

Fig. 14-76: Task Properties

Stop1



The Stop1 icon is used to halt motion on one or all axes used in a task. It will return the drives to an AH (Drive halt) state. Placing a Stop1 icon on a task or subroutine workspace automatically displays a *Stop Setup* window.



Stop.tif

Fig. 14-77: Stop Setup

Rexroth Bosch Group

Integers and labels can be selected from the VM Data Table by clicking the browse button to the right of the **Axis** field.

Coordinated Abort and **Coordinated Stop** decelerate motion on path, stopping all axes associated with the coordinated motion. Selecting Coordinated Abort or Coordinated Stop enables a pop-down menu permitting selection of one of the four control tasks.

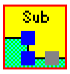
Restarting motion after a Coordinated Abort requires toggling the Cycle/Start bit of the associated Task Control register. Motion stopped using a Coordinated Stop may be resumed with a Go1 icon, although resuming timed events that are programmed for motion at operating speed may result in events occurring at unexpected times.

Non-Coordinated Abort and **Non-Coordinated Stop** stops motion on the specified axis by decelerating the axis to zero velocity using the currently programmed rate. The axis to stop may be specified by a valid integer constant, variable, global variable or an equivalent label. Specifying a "-1" stops all axes in the task.

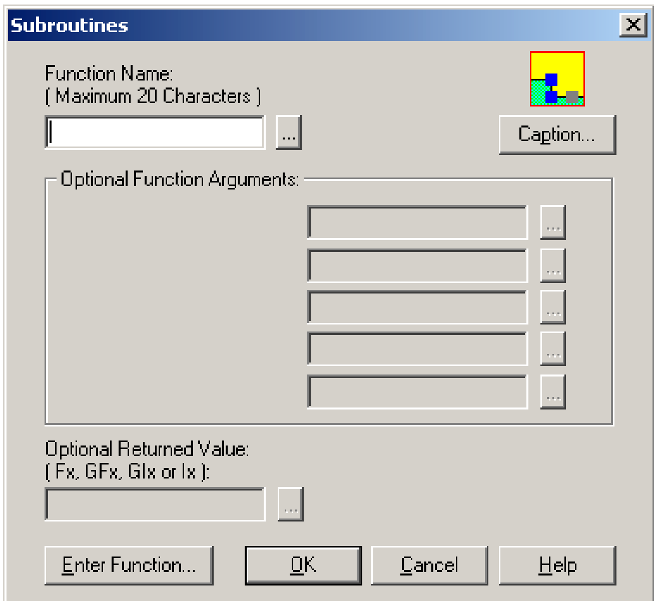
Note: After an **Abort**, all queued events and the "look-ahead" motion calculated by the path planner are lost and the current move is aborted. The target position is set equal to the current feedback position (if A-0-0164, bit 3 = 0). After a **Stop**, both queued events and the calculated path are retained. The target position is not set equal to the feedback position. When the axis is again enabled, it will complete the last commanded move

---

Sub1



The Sub1 icon is used to invoke (call) a subroutine within the program flow of a task, other subroutine or event function. Placing the Sub1 icon in an icon program workspace opens the *Subroutines* window. The **Optional Function Argument** and **Optional Return Value** fields are initially grayed out.



Subroutine.tif

Fig. 14-78: Subroutine Setup

The user can assign a label in the **Function Name** field that will be used to identify the subroutine. Selecting the browse button to the right of the **Function Name** field allows the user to choose an existing subroutine.

To create the icon program for the subroutine, click the ***Enter Function…*** button after naming the subroutine. The *Subroutines* window closes and a new icon program workspace opens containing a Start and Finish1 icon. The user can then select and place the desired icons to create the program that will run when the Sub1 icon is encountered in the calling program flow. If the ***OK*** button is clicked before the ***Enter Function…*** button, then the *Subroutines* window closes and the Sub1 icon appear in the program flow. The user can then return at a future time to program the subroutine.

Note: Subroutine can be viewed by any of the following methods:

- Click the ***Enter Function…*** button in the Sub1 icon.
- Select ***View*** ⇒ ***Subroutine*** from VisualMotion Toolkit's main menu, highlight the desired subroutine and click the ***View/Edit…*** button.
- Select the subroutine name from the Project Navigator
- Browsing for Icon flow

The name assigned to the subroutine will appear in the *Project Navigator* window. Subroutines created for the Initialization task appear under **Initialization Task**. Subroutines created for a main task, other subroutines or event functions will all appear under **Subroutines**.

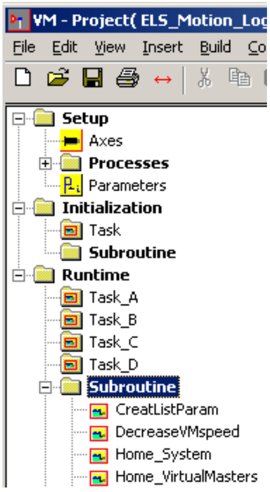


Fig. 14-79: Subroutine Names in Project Navigator Window

Note: VisualMotion 11 has an Initialization task separate from the four main tasks (A-D). Subroutines created for the Initialization task are unique to that task and should not be called from a main task, subroutine or event function. Likewise, main task subroutines should not be called from the Initialization task.

Optional Function Arguments

Function arguments are stack-based variables whose values exist only while in the subroutine (initialization or standard) where they are declared. The ***Define Function Arguments*** button in the Start1 icon allows the user to define up to 5 function argument names and assign data types to them. These function arguments are variables (float, integer, ABS index or REL index). Refer to Define Function Arguments (Start1 icon) on page 14-75 for details.

Passing Constants

When passing constant values to a subroutine with defined function arguments, the value can be used and/or modified. When the subroutine ends, the modified value is not returned to the caller unless the Optional Returned Value field is used.

Passing Variables

When passing a variable value to a subroutine with defined function arguments, a copy of the variable's current value is passed in to the subroutine. If the passed value is modified while in the subroutine, the original variable existing in the calling task or subroutine is not affected.

The **Optional Function Argument** fields become active in the *Subroutines* window when function arguments are defined in the Start1 icon of the subroutine. The function argument names used in the Start1 icon will appear in the *Subroutines* window after they are defined. The value that will be passed to the function argument is entered in the **Optional Function Arguments** fields.

The **Optional Returned Value** field becomes active in the *Subroutines* window when a value is entered in the **Optional return value** field in the Finish1 icon of the subroutine. The value returned from the Finish1 icon will be written to the variable entered in the **Optional Returned Value** field of the *Subroutines* window.

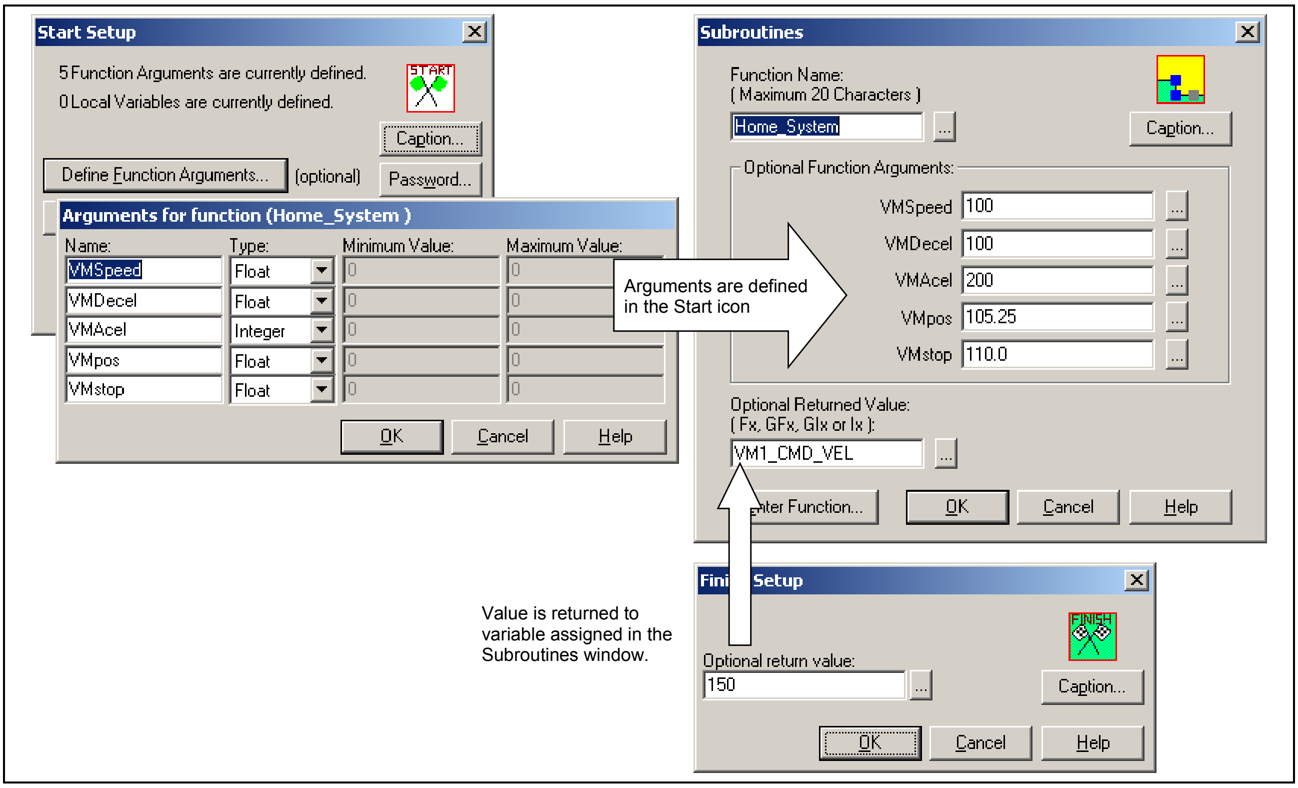
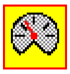


Fig. 14-80: Sub1 Icon – Optional Function Arguments

Function arguments are only allowed in the Start1 icon of a subroutine. Start1 icons in any task or event function will have this feature grayed out. However, an optional return value may be passed back to a task that calls the subroutine. The **Optional Return Value** in the Finish1 icon of a subroutine is a good way of getting a value from a subroutine.

Veloc (Velocity)



The Velocity icon is used to specify a rate for motion on a single non-coordinated axis. The axis may be specified by a valid integer constant, variable, global variable or an equivalent label. The velocity may be entered as a float constant, variable, global variable or an equivalent label. Clicking the browse button to the right of any field opens the VM Data Table.

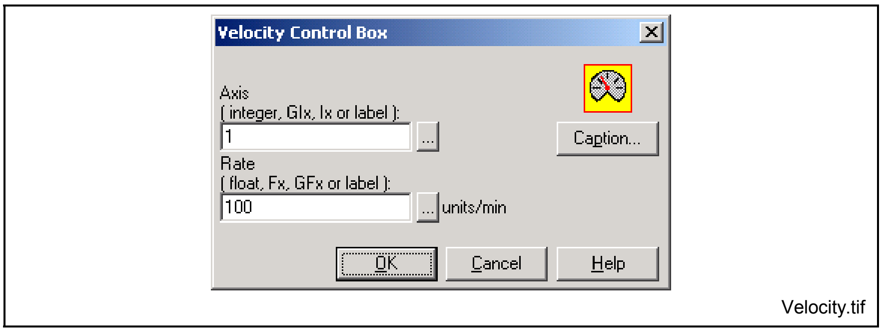
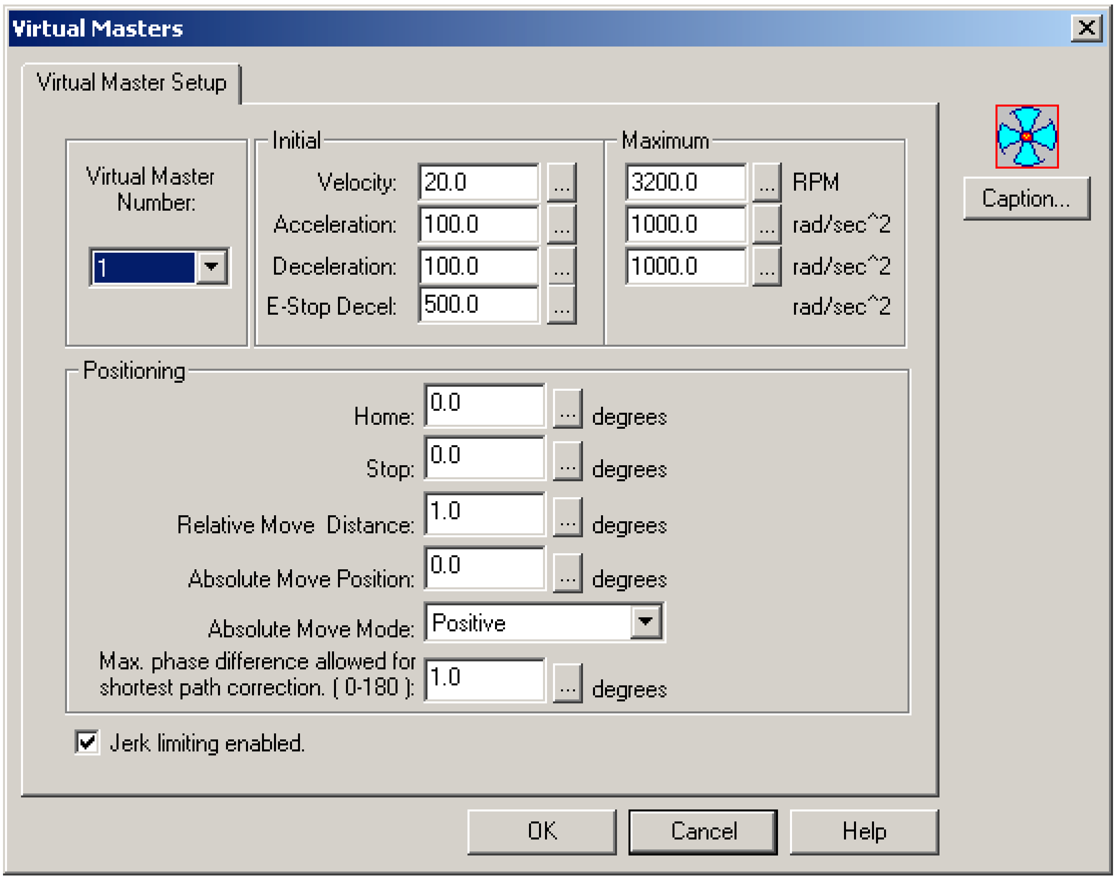


Fig. 14-81: Velocity Setup

VM1 (Virtual Master)

A Virtual Master configuration must first be declared under ***Setup*** ⇒ ***Processes*** ⇒ ***ELS*** ⇒ ***Virtual Masters***. Afterwards, an VM 1 icon can be placed in the Initialization task to modify an existing Virtual Master setup. VisualMotion supports 2 Virtual Masters in a G*P 11 project.

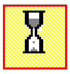


Virtual_Master_Init_Val.tif

Fig. 14-82: Configure Initial Values for Virtual Master

Refer to section 4.5, *Setup ELS Process*, in volume 1 for details. For specific information on assigning initial values, refer to section 6.1 in volume 1.

Wait1



The Wait1 icon is used to hold the execution of the program flow at the Wait1 icon until a specified condition has been satisfied. The condition may be related to a single axis' position, time, I/O state, or path planner state (for coordinated motion). Placing a Wait1 icon on a task or subroutine workspace automatically displays a *Wait Control Box* window.

Axis in Position pauses program execution until the specified axis reaches the in position window of the associated drive. The axis may be specified by a valid integer constant variable (Ix) global variable (Gix) or an equivalent label.

If a "-1" is entered, program execution will wait until all axes assigned to the task are within their respective position windows before continuing.

Axis at Position pauses program execution until the specified axis reaches a specified position for the associated drive.

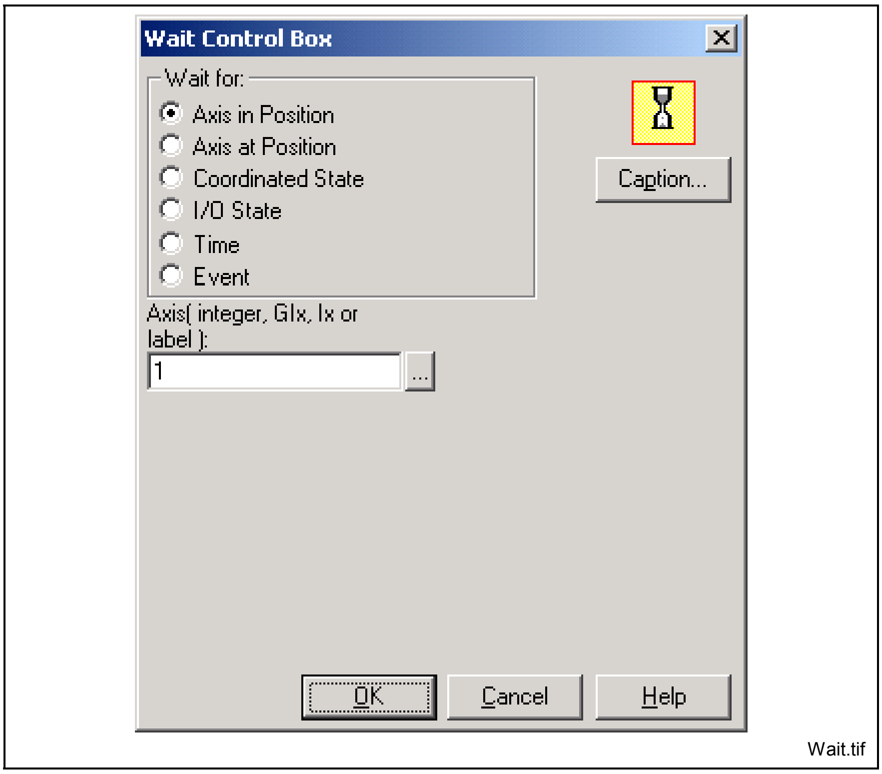
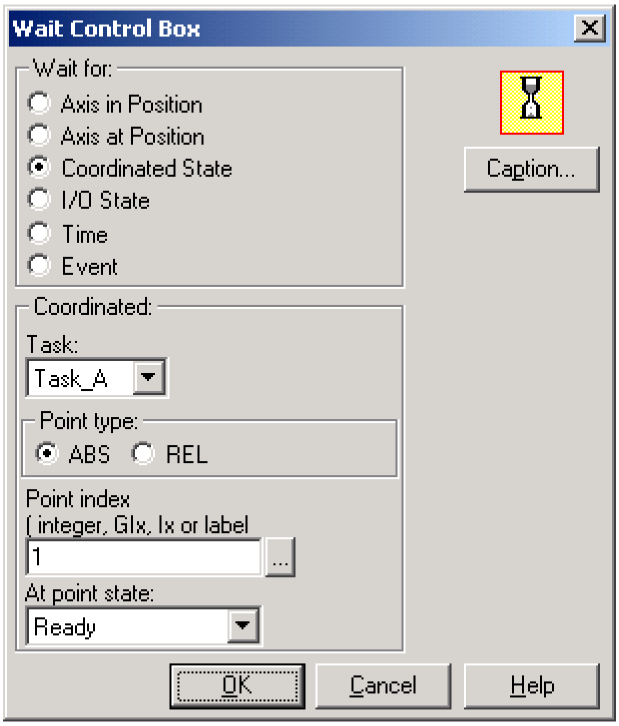


Fig. 14-83: Wait Setup

Selecting **Coordinated State** pauses program execution and tests the state of the path planner for the specified point until the planner enters the selected processing state. Coordinated Waits are specific to each task. You can wait in one task for the coordinated state of another



Wait_Coord.tif

Fig. 14-84: Wait Icon Setup for Coordinated State

The **At point state** field provides the seven path planner test states listed below:

- **Ready** - Path planner has processed and queued the specified point.
- **Accelerating** - Task's coordinated motion is accelerating into the segment ending at the specified point.
- **Constant Velocity** - Task's coordinated motion is traversing the segment ending at the specified point.
- **Blending** - Task's coordinated motion is traversing the blend segment calculated for the specified point.
- **In Decel to Target** - Task's coordinated motion is in deceleration in the segment ending at the specified point.
- **At Controlled Stop** - Task's coordinated motion is decelerating on the segment ending at the specified point after a commanded stop.
- **Done** - Task's coordinated motion has completed for the specified point.

I/O State pauses program execution until the specified I/O condition for the specified I/O register is satisfied. The I/O register is specified by entering a valid I/O register ID number or equivalent label. Clicking the browse button to the right of any field opens the VM Data Table.

The contents of the specified register may be "anded" with a bit mask for the register contents. The bit mask allows "masking out" unwanted bits (by specifying "0" in the bit position in the mask), and permits the on/off condition to be effected by more than a single bit (by specifying "1" to enable the bit).

The **I/O State** radio buttons determine how the wait condition is satisfied. I/O State "on" (high or "1") requires that all the enabled bits are logical one. I/O State "Off" (low or "0") requires that all enabled bits are zero.

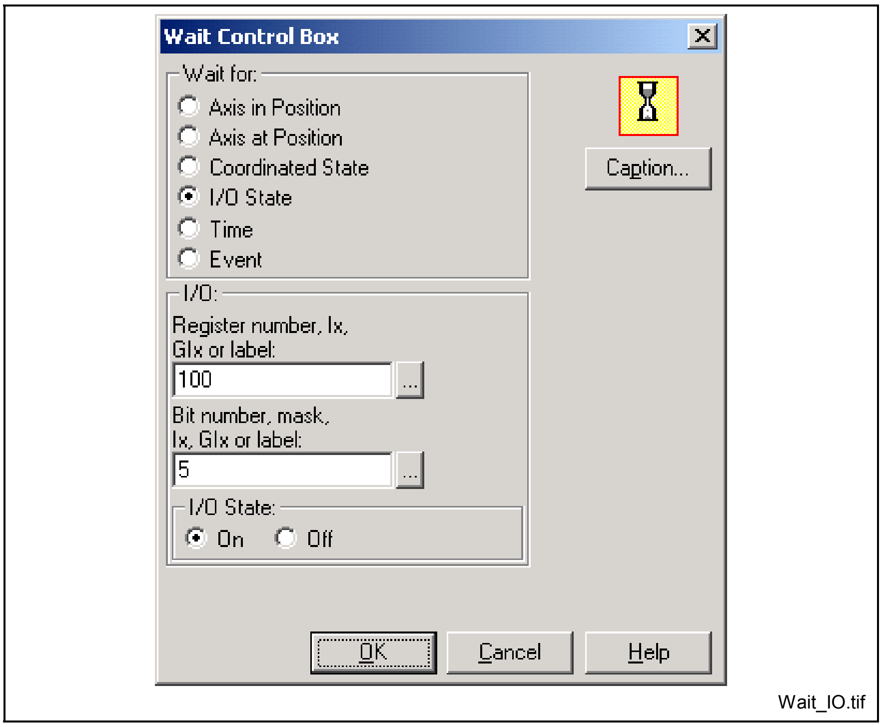


Fig. 14-85: Wait Setup for I/O state

Time pauses program execution for a specified time delay. Enter the number of milliseconds in the **Time Delay** field. The delay may be specified by an integer constant, variable (Ix), global variable (GIx), or an equivalent label. Clicking the browse button to the right of any field opens the VM Data Table.

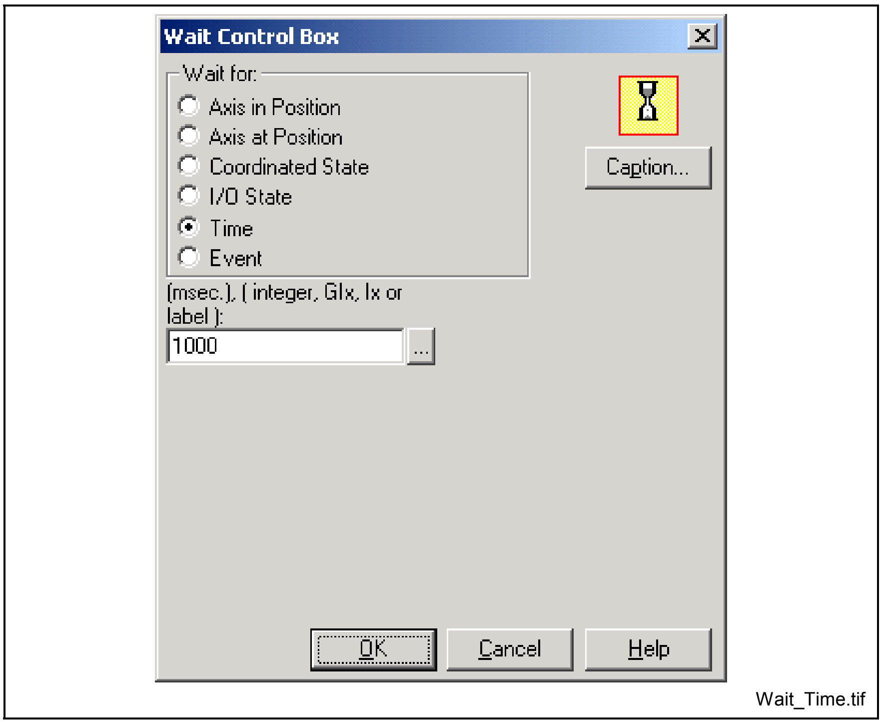
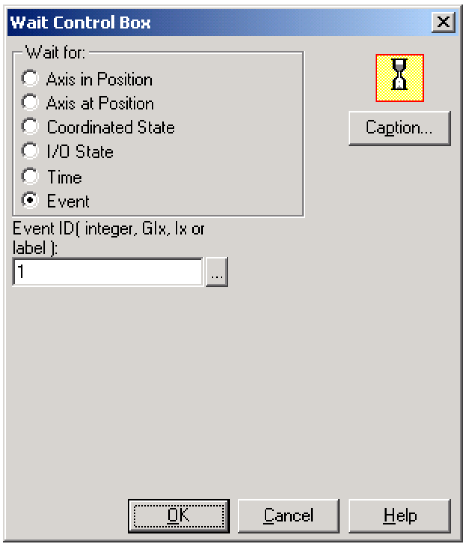


Fig. 14-86: Wait Setup for Time

Event pauses program execution until the specified event has completed. The event may be specified by an integer constant, variable (Ix), global variable (GIx), or an equivalent label. The VM Data Table can be displayed by clicking on the button to the right of the **Event ID** field.



Wait_Event.tif

Fig. 14-87: Wait for Event

15 Parameters

15.1 Overview

Rexroth VisualMotion controls provide parameters to adapt to a specific application. Basic communication and initialization parameters for motion, logic, and digital drives must be entered before it is possible to operate or program a system.

System-builders must modify certain parameters describing the mechanical characteristics of their unique application. Parameters specify machine limitations, such as maximum velocity and acceleration. Other parameters are used to specify the mechanical characteristics of the system, such as the ratio between motor revolutions and shaft rotation and slide travel.

VisualMotion structures parameters into four classes:

- Control (C-parameters)
- Task (T-parameters)
- Axis (A-parameters)
- Drive (S-standard Sercos and P-product specific)

Control parameters contain setup and status parameters for the control. Task and Axis parameters are associated with the user program. Digital drive parameters are stored in each Sercos-compatible digital drive and are necessary for configuring a motion system.

Note: Only control, task and axis parameters will be described in this chapter. Rexroth digital drives uses S (Sercos) and P (product specific) parameters for setting up and configuring each drive. Refer to the respective *Digital Drive* documentation or *Drive Help System* for a description of these drive parameters.

Control, task, axis, and drive parameters can be accessed via a serial, Ethernet, or PCI interface using any of the following methods:

- Rexroth VisualMotion Toolkit
- Windows™ based HMI software
- PLC

Some parameters may be accessed by the user program through a **Param1** icon. VisualMotion also allows access to specific groups of parameters for editing and archiving.

Refer to *Chapter 14, Icon Descriptions,* and *Chapter 19, Communication Protocol,* for details.

15.2 Parameter Identification

All parameters in VisualMotion have a unique **ID**entification **N**umber (IDN) and are displayed using the Sercos format:

Sercos Format:

```
## X-s-nnnn
|  |  | |_ 4 digit parameter number
|  |  |___ Set: 0 or 7
|  |______ Class: Parameter type
|_________ Class Number
```

Example: 01 S-0-0001
Description: NC cycle time for drive 1

Legend:

##: Class Number
(01) fixed for C parameters
(01-64) Sercos drive address for A,S, and P parameters
(01-04)Task:A=01,B=02,C=03,D=04

X : Class: Parameter type
C – Control
T – Task
A – Axis
S – Sercos drive parameter
P – Product specific parameter

s : Certain S and P drive parameters belong to two different sets
0:set parameters can be modified
7:set parameters are read only

n : 4 digit parameter number
(0000 – 4095)

Parameter Attributes

The following attributes for each parameter are provided:

- Data length
- Data type
- Display format
- Units
- Minimum value
- Maximum value
- Default value
- Access (read / write)

List Parameter Access

The simultaneous access (reading/writing) of more than one list parameter by means of any of the following processes can cause unwanted system errors which can interrupt the execution of the running program:

Process	Details
parameter transfer icon	reading or writing data
parameter initialization table	
fieldbus non-cyclic channel	
serial / Ethernet/ PCI communication	
SIS communication	for standard services (0x10, 0x11, 0x1E, 0x1F) for VM services (0xC3, 0xC4)
ASCII communication	including standard and block transfers
function block	only when accessing a list parameter such as MSV_ReadList, CAM building FB's, etc.
CAM build icon	when accessing drive CAMs
Calc icon	when accessing drive PLS

Table 15-1: Processes Accessing List Parameters

Class "C" Parameters

Class "C" control parameter types include the system setup and status parameters for VisualMotion. Certain status parameters are stored on the control in non-volatile memory, while other status parameters are only temporary indicators and are lost when the system is powered off.

Class "T" Parameters

Class "T" task parameter types include setup and status information for each task. Tasks A-D are represented by a corresponding numbers 01-04 respectively.

Class "A" Parameters

Class "A" axis parameter types include the parameters used to configure and display information about the each axis. A maximum of 64 different sets of parameters are possible. A set corresponding to one of the eight axes is identified by a single digit in the range 01-64.

Class "S" and "P" Parameters

Class "S" (Sercos) drive parameters are accessed through the Sercos Service Channel.

Class "P" (Product specific) drive parameters include the parameters required to configure and operate Rexroth's digital drives. These parameters are unique to Rexroth drives and are necessary for proper communication between the drives and external devices.

Most servo and position loop parameters are contained in Sercos compatible digital drives. These parameters include Feed Constant, Kv Factor, In-Position Window, Monitor Window, and all homing parameters. Acceleration, Deceleration, and Jerk are defined in the user program and limits for these are set in the drive.

15.3 Parameter Transfer

Parameter transfer capability is built into VisualMotion's programming language. The **Param1** icon is used for parameter transfers and command execution. It allows the monitoring of system values and Sercos command execution.

All control and drive parameters of the float type may be read into a float variable. All other parameters, unless they are a String or List type, can be read into an integer variable. VisualMotion issues a runtime error if values are transferred to/from the wrong type of variable. The parameters with "read / write in any Sercos phase" access may be changed by a user program.

15.4 Sercos Drive Telegram Utility

The *Parameter Overview* tool contains a Sercos Drive Telegram Utility used to configure a drive's AT (drive telegram) and MDT (master data telegram). The AT and MDT are used to cyclically exchange parameter data between drives and control every Sercos cycle. The AT is sent from each drive to the control and the MDT is sent from the control to each active drive on the system.

For detailed information on *Sercos drive telegram*, refer to the *Functional Description*, *section 7.3*.

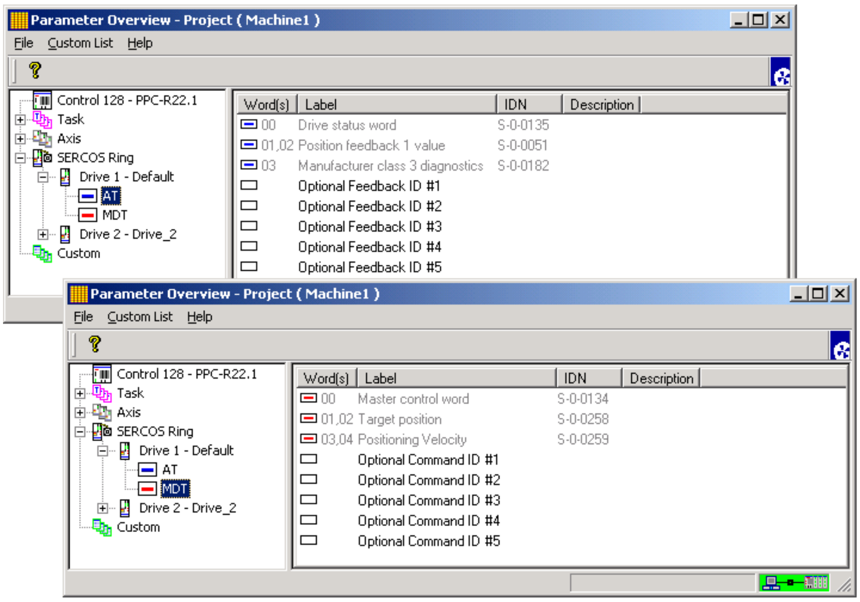


Fig. 15-1: Sercos Drive AT and MDT

15.5 Parameter Types

The information provided in this chapter will focus on control, task, and axis parameter types. Drive parameters vary based on the type of digital drive and firmware used. Refer to the relevant *Rexroth Digital Drive* manual for specific drive parameter descriptions.

Control Parameters - Class C

This section provides a list of all control parameters that are available in the system. They are grouped by functionality and indicate the supported control firmware type.

System Setup Parameters

Parameter Number	Description	Valid for GPP 11	Valid for GMP 11
C-0-0001	Language Selection	X	X
C-0-0002	Control Address	X	X
C-0-0003	X10 Programming Port Setup	X	X
C-0-0004	X16 Communication Port Setup	X	X
C-0-0005	Communication Protocol Selection	X	X
C-0-0008	System Control Word Status	X	X
C-0-0009	Error Reaction Mode	X	X
C-0-0010	System Options	X	X
C-0-0011	System Control Word	X	X
C-0-0012	X16 Communication Port Device Type	X	X
C-0-0013	X10 Programming Port Mode	X	X
C-0-0014	X16 Communication Port Mode	X	X
C-0-0016	Communication Time-out Period	X	X
C-0-0017	X10 Programming Port Password	X	X
C-0-0018	X16 Communication Port Password	X	X
C-0-0020	Transmitter Fiber Optic Length	X	X
C-0-0021	User Watchdog Timer	X	X
C-0-0022	User Watchdog Task ID	X	X

Table 15-2: System Setup Parameters

Jogging and Display Parameters

Parameter Number	Description	Valid for GPP 11	Valid for GMP 11
C-0-0042	World Large Increment	X	X
C-0-0043	World Small Increment	X	X
C-0-0045	World Fast Jog Speed	X	X
C-0-0046	World Slow Jog Speed	X	X
C-0-0052	Axis Large Increment	X	X
C-0-0053	Axis Small Increment	X	X
C-0-0055	Axis Fast Jog Velocity	X	X
C-0-0056	Axis Slow Jog Velocity	X	X

Table 15-3: Jogging and Display Parameters

Program Integer and Float Data Transfer Range

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-0070	Data Transfer Start of Program Floats	X	X
C-0-0071	Data Transfer End of Program Floats	X	X
C-0-0072	Data Transfer Start of Program Integers	X	X
C-0-0073	Data Transfer End of Program Integers	X	X

Table 15-4: Program Integer and Float Transfer Values Parameters

Global Variable Command Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-0080	Maximum Number of Global Integers	X	X
C-0-0081	Maximum Number of Global Floats	X	X
C-0-0082	Save Global Variables Command	X	X
C-0-0083	Save Global Variables Status	X	X

Table 15-5: Global Variable Command Parameters

Program Management Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-0090	Download Block Size	X	X
C-0-0091	Total Program Memory	X	X
C-0-0092	Available Program Memory	X	X
C-0-0093	Contiguous Program Memory	X	X
C-0-0094	Maximum Executable Program Size	X	X
C-0-0098	Initialization Delay	X	X
C-0-0099	Minimum Sercos Cycle Time	X	X

Table 15-6: Program Management Parameters

System Status Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-0100	Control Firmware Version	X	X
C-0-0101	Control Hardware Version	X	X
C-0-0102	Control Version Date	X	X
C-0-0103	Allowable Drive Address Range	X	X
C-0-0104	Bootloader Firmware Version	X	X
C-0-0120	Operating Mode	X	X
C-0-0121	Sercos Communication Phase	X	X
C-0-0122	Diagnostic Message	X	X
C-0-0123	Diagnostic Code	X	X
C-0-0124	Extended Diagnostic	X	X
C-0-0125	System Timer Value	X	X
C-0-0126	Date and Time	X	X
C-0-0127	Current PPC-R22.1 Temperature	X	X
C-0-0128	Elapsed Time Operational Counter	X	X
C-0-0142	Card Label String	X	X

Table 15-7: System Status Parameters

Save CAM Built to Flash Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-0166	Save Built CAM to Flash ID	X	X
C-0-0167	Save CAM Built to Flash Command	X	X
C-0-0168	Save Built CAM to Flash Status	X	X

Table 15-8: Save CAM Built to Flash Parameters

NVRAM and Compact Flash Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-0170	NVRam backup command	X	X
C-0-0171	NVRam backup status	X	X
C-0-0172	Compact flash backup command	X	X
C-0-0173	Compact flash backup status	X	X
C-0-0174	Compact flash restore command	X	X
C-0-0175	Compact flash restore status	X	X
C-0-0176	Compact flash Date / Time Backup	X	X
C-0-0177	Compact flash Firmware Version	X	X

Table 15-9: NVRAM and Compact Flash Parameters

Control Processor Usage Status Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-0200	Current Load Due To Motion	X	X
C-0-0201	Peak Load Due To Motion	X	X
C-0-0202	Current Load Due To IO	X	X
C-0-0203	Peak Load Due To IO	X	X

Table 15-10: Control Processor Usage Status Parameters

Link Ring Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-0300	Link Ring Control Word	X	X
C-0-0301	Link Ring Primary Fiber Optic Length	X	X
C-0-0302	Link Ring Secondary Fiber Optic Length	X	X
C-0-0303	Link Ring MDT Error Counter	X	X
C-0-0304	Link Ring Node Address	X	X
C-0-0305	Link Ring Cycle Time	X	X

Table 15-11: Link Ring Parameters

Ethernet 1 Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-0400	Ethernet 1 IP Address	X	
C-0-0401	Ethernet 1 Subnet Mask	X	
C-0-0402	Ethernet 1 Gateway IP Address	X	
C-0-0403	Ethernet 1 Duplex Mode	X	
C-0-0404	Ethernet 1 Access Control	X	
C-0-0405	Ethernet 1 Interface Password	X	
C-0-0406	Hardware ID of Ethernet 1 Interface	X	
C-0-0407	Ethernet 1 Interface Firmware Version	X	
C-0-0408	Ethernet 1 Driver Version	X	
C-0-0409	MAC Address of Ethernet 1 Interface	X	
C-0-0410	Fieldbus slave Cyclic Channel Status	X	

Table 15-12: Ethernet 1 Parameters

Ethernet 2 Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-0411	Ethernet 2 IP Address	X	
C-0-0412	Ethernet 2 Subnet Mask	X	
C-0-0413	Ethernet 2 Gateway IP Address	X	
C-0-0415	Ethernet 2 Access Control	X	
C-0-0416	Ethernet 2 Interface Password	X	
C-0-0417	Hardware ID of Ethernet 2 Interface	X	
C-0-0418	Ethernet 2 Interface Firmware Version	X	
C-0-0419	Ethernet 2 Driver Version	X	
C-0-0420	MAC Address of Ethernet 2 Interface	X	

Table 15-13: Ethernet 2 Parameters

User and Initialization Task Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-0503	Initialization Task Instruction Rate	X	X
C-0-0522	Init. Task Diagnostic Message	X	X
C-0-0523	Init. Task Status Message	X	X
C-0-0530	Init. Task Current Instr. Pointer	X	X
C-0-0531	Init. Task Current Instruction	X	X
C-0-0532	Init. Task Instr. Pointer at Error	X	X
C-0-0533	Init. Task Composite Instr. Pointer	X	X
C-0-0535	Init. Task Current Subroutine	X	X
C-0-0536	Init. Task Stack Variable Data	X	X
C-0-0537	Init. Task Subroutine Breakpoint	X	X

Table 15-14: User and Initialization Task Parameters

BTC06 Teach Pendant Parameters

Parameter Number	Description	Valid for GPP 11	Valid for GMP 11
C-0-0801	Pendant Protection Level 1 Password	X	X
C-0-0802	Pendant Protection Level 2 Password	X	X
C-0-0803	Pendant User Accessible Floats Section	X	X
C-0-0804	Pendant User Accessible Integers	X	X
C-0-0805	Pendant Start of User Accessible Reg.s	X	X
C-0-0806	Pendant End of User Accessible Reg.	X	X
C-0-0807	Pendant Password Timeout	X	X
C-0-0810	TPT message and prompt control word	X	X
C-0-0811	User Task Controlled Menu ID for TPT	X	X
C-0-0812	User Task Controlled Task ID for TPT	X	X
C-0-0813	User Task Controlled Axis Number TPT	X	X
C-0-0814	TPT Data Transaction Word	X	X

Table 15-15: Pendant Parameters

System Memory Parameters

Parameter Number	Description	Valid for GPP 11	Valid for GMP 11
C-0-0993	Software Reset for Control	X	X
C-0-0994	Shutdown Command for Flash Programming	X	X
C-0-0996	Clear Program and Data Memory	X	X
C-0-0997	Clear Diagnostic Log	X	X

Table 15-16: System Memory Parameters

Integrated PLC Parameters

Parameter Number	Description	Valid for GPP 11	Valid for GMP 11
C-0-1600	Integrated PLC: Timeslice of Sercos Cycle Time	X	X
C-0-1601	Integrated PLC: Configuration	X	X
C-0-1602	Integrated PLC: Control word status	X	X
C-0-1603	Integrated PLC: Control word	X	X
C-0-1604	Integrated PLC: Status word	X	X
C-0-1605	Integrated PLC: Project name	X	X
C-0-1610	ELS System Association	X	X
C-0-1611	Integrated PLC: Diagnostic Code	X	X
C-0-1612	Integrated PLC: Diagnostic Message	X	X
C-0-1613	Integrated PLC: Extended Diagnostic	X	X
C-0-1620	Integrated PLC: POU Error Log List	X	X
C-0-1621	Integrated PLC: POU Error Log List Options	X	X
C-0-1638	Integrated PLC: SysLibDirect Read Access Parameters	X	X
C-0-1639	Integrated PLC: SysLibDirect Write Access Parameters	X	X

Table 15-17: System Memory Parameters

System Parameter Lists

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-2000	List of All Parameters	X	X
C-0-2001	List of Required Parameters	X	X
C-0-2002	List of invalid A-, C- and T- parameters	X	X
C-0-2010	List of Sercos Devices	X	X
C-0-2011	List of Sercos Drives	X	X
C-0-2012	List of Sercos I/O Stations	X	X
C-0-2013	I/O Configuration List	X	X
C-0-2016	List of Virtual axes	X	X
C-0-2017	I/O User Configuration List	X	X
C-0-2019	Beckoff I/O Configuration List	X	X
C-0-2020	Diagnostic Log List	X	X
C-0-2021	Diagnostic Log Options	X	X
C-0-2022	Probe Exception Handler	X	X

Table 15-18: System Parameter Lists

Oscilloscope Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-2501	Oscilloscope signal 1 type	X	X
C-0-2502	Oscilloscope signal 2 type	X	X
C-0-2503	Oscilloscope signal 3 type	X	X
C-0-2524	Oscilloscope signal 4 type	X	X
C-0-2504	Oscilloscope signal 1 id number	X	X
C-0-2505	Oscilloscope signal 2 id number	X	X
C-0-2506	Oscilloscope signal 3 id number	X	X
C-0-2525	Oscilloscope signal 4 id number	X	X
C-0-2507	Oscilloscope signal 1 axis number	X	X
C-0-2508	Oscilloscope signal 2 axis number	X	X
C-0-2509	Oscilloscope signal 3 axis number	X	X
C-0-2526	Oscilloscope signal 4 axis number	X	X
C-0-2510	Oscilloscope sample rate	X	X
C-0-2511	Oscilloscope signal 1 list	X	X
C-0-2512	Oscilloscope signal 2 list	X	X
C-0-2513	Oscilloscope signal 3 list	X	X
C-0-2527	Oscilloscope signal 4 list	X	X
C-0-2514	Oscilloscope sample count	X	X
C-0-2515	Oscilloscope trigger post-count	X	X
C-0-2516	Oscilloscope trigger type	X	X
C-0-2517	Oscilloscope trigger id number	X	X
C-0-2518	Oscilloscope trigger axis or mask	X	X
C-0-2519	Oscilloscope trigger level or mask	X	X
C-0-2520	Oscilloscope trigger mode	X	X
C-0-2521	Oscilloscope trigger source	X	X
C-0-2522	Oscilloscope trigger control word	X	X
C-0-2523	Oscilloscope trigger status word	X	X

Table 15-19: Oscilloscope Parameters

Fieldbus/PLC Interface Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-2600	Fieldbus/PLC Mapper (cyclic channel) To PLC	X	X
C-0-2601	Fieldbus/PLC Mapper (cyclic channel) From PLC	X	X
C-0-2607	Multiplex Control Word	X	X
C-0-2608	Multiplex Status Word	X	X
C-0-2611	Fieldbus/PLC Cyclic Channel: Current number of misses	X	X
C-0-2612	Fieldbus/PLC Cyclic Channel: Peak number of misses	X	X
C-0-2613	Fieldbus/PLC Cyclic Channel: Timeout counter	X	X
C-0-2630	Fieldbus Slave Device Address	X	
C-0-2631	Fieldbus Parameter/PCP Channel Length	X	
C-0-2632	Fieldbus/PLC Multiplex Method	X	X
C-0-2633	Fieldbus Baud Rate (DeviceNet only)	X	
C-0-2635	Fieldbus/PLC Error Reaction	X	X
C-0-2636	Fieldbus/PLC Word Swap	X	X
C-0-2637	Fieldbus Slave/PLC Firmware Version	X	X
C-0-2638	Fieldbus Available Cyclic IN Parameters	X	X
C-0-2639	Fieldbus Available Cyclic OUT Parameters	X	X

Table 15-20: Fieldbus Interface Parameters

PLC Interface Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-2640	PLC Connection Options		X
C-0-2641	PLC Input Register List	X	X
C-0-2642	PLC Output Register List	X	X
C-0-2643	PLC Lifecounter Check: Number of Retries		X
C-0-2644	PLC Lifecounter Check: Current Number of Misses		X
C-0-2645	PLC Lifecounter Check: Peak Number of Misses		X
C-0-2646	PLC Lifecounter Check: Number of Timeouts		X
C-0-2651	PLC Register Channel: Current number of misses		X
C-0-2652	PLC Register Channel: Peak number of misses		X
C-0-2653	PLC Register Channel: Timeout counter		X

Table 15-21: PLC Interface Parameters

Encoder Interface Card Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-2800	MEC Hardware Version	X	X
C-0-2801	MEC Firmware Version	X	X
C-0-2802	MEC Configuration	X	X
C-0-2805	MEC Error Type	X	X
C-0-2806	MEC Error Group	X	X
C-0-2807	MEC Error Number	X	X
C-0-2810	MEC Encoder Detection	X	X
C-0-2811	MEC Encoder Type Detection	X	X
C-0-2815	MEC Direction of Encoder 1	X	X
C-0-2816	MEC Direction of Encoder 2	X	X
C-0-2817	MEC Encoder 1 Resolution	X	X
C-0-2818	MEC Encoder 2 Resolution	X	X

Table 15-22: Encoder Interface Card Parameters

Option Card PLS Interface Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-2901	PLS1 Start Output Register	X	X
C-0-2902	PLS1 Start Mask Register	X	X
C-0-2903	PLS1 Build Table Command	X	X
C-0-2904	PLS1 Build Table Status	X	X
C-0-2905	PLS1 Activate Command	X	X
C-0-2906	PLS1 Activate Status	X	X
C-0-2907	PLS1 Error Code	X	X
C-0-2908	PLS1 Extended Error Code	X	X
C-0-2909	PLS1 Hardware ID	X	X
C-0-2910	PLS1 Software ID	X	X
C-0-2920	PLS1 Switch On List	X	X
C-0-2921	PLS1 Switch Off List	X	X
C-0-2922	PLS1 Switch Output List	X	X
C-0-2930	PLS1 Output Master List	X	X
C-0-2931	PLS1 Output Lead Time List	X	X
C-0-2932	PLS1 Output Lag Time List	X	X
C-0-2933	PLS1 Output One Shot (PT) List	X	X
C-0-2934	PLS1 Output Mode List	X	X
C-0-2935	PLS1 Output Direction List	X	X
C-0-2936	PLS1 Output Hysteresis List	X	X
C-0-2940	PLS1 Master Type List	X	X
C-0-2941	PLS1 Master Number List	X	X
C-0-2942	PLS1 Master Encoder List	X	X
C-0-2943	PLS1 Master Phase Offset List	X	X

Table 15-23: Option Card PLS Interface Parameters

CAM Table Parameters

Parameter Number	Description	Valid for GPP 11	GMP 11
C-0-3100	CAM Tags	X	X
C-0-3101	CAM Table 1	X	X
C-0-3102	CAM Table 2	X	X
C-0-3103	CAM Table 3	X	X
C-0-3104	CAM Table 4	X	X
C-0-3105	CAM Table 5	X	X
C-0-3106	CAM Table 6	X	X
C-0-3107	CAM Table 7	X	X
C-0-3108	CAM Table 8	X	X
C-0-3109	CAM Table 9	X	X
consecutively through			
C-0-3140	CAM Table 40	X	X
C-0-3141	CAM Type	X	X

Table 15-24: CAM Table Parameters

Position Monitor Group Parameters

Parameter Number		Description	Valid for GPP 11	GMP 11
C-0-3201	C-0-3211	PMG # Maximum Allowed Deviation Window where # = group numbers 1-8	X	X
C-0-3221	C-0-3231			
C-0-3241	C-0-3251			
C-0-3261	C-0-3271			
C-0-3202	C-0-3212	PMG # List of Axes	X	X
C-0-3222	C-0-3232			
C-0-3242	C-0-3252			
C-0-3262	C-0-3272			
C-0-3203	C-0-3213	PMG # List of Position Offsets	X	X
C-0-3223	C-0-3233			
C-0-3243	C-0-3253			
C-0-3263	C-0-3273			
C-0-3204	C-0-3214	PMG # Current Peak Group Deviation	X	X
C-0-3224	C-0-3234			
C-0-3244	C-0-3254			
C-0-3264	C-0-3274			
C-0-3205	C-0-3215	PMG # Maximum Deviation	X	X
C-0-3225	C-0-3235			
C-0-3245	C-0-3255			
C-0-3265	C-0-3275			
C-0-3206	C-0-3216	PMG # Configuration	X	X
C-0-3226	C-0-3236			
C-0-3246	C-0-3256			
C-0-3266	C-0-3276			

Table 15-25: Position Monitoring Group Parameters

Task Parameters - Class T

This section provides a list of all task parameters that are available in the system. They are grouped by functionality and indicate the supported control firmware type.

Task Setup Parameters

Parameter Number	Description	Valid for GPP 11	Valid for GMP 11
T-0-0001	Task Motion Type	X	X
T-0-0002	Task Options	X	X
T-0-0003	Task Instruction Rate	X	X

Table 15-26: Task Setup Parameters

Coordinated Motion

Parameter Number	Description	Valid for GPP 11	Valid for GMP 11
T-0-0005	World Position Units	X	X
T-0-0010	Kinematics Number	X	X
T-0-0011	Coordinated X-axis	X	X
T-0-0012	Coordinated Y-axis	X	X
T-0-0013	Coordinated Z-axis	X	X
T-0-0020	Maximum Path Speed	X	X
T-0-0021	Maximum Acceleration	X	X
T-0-0022	Maximum Deceleration	X	X
T-0-0023	Look Ahead Distance	X	X
T-0-0024	Velocity Override	X	X
T-0-0025	Maximum Jog Increment	X	X
T-0-0026	Maximum Jog Velocity	X	X
T-0-0027	Path Smoothing Filter Constant	X	X
T-0-0050	Kinematics Value 1	X	X
T-0-0051	Kinematics Value 2	X	X
T-0-0052	Kinematics Value 3	X	X
T-0-0053	Kinematics Value 4	X	X
T-0-0054	Kinematics Value 5	X	X
T-0-0055	Kinematics Value 6	X	X
T-0-0056	Kinematics Value 7	X	X
T-0-0057	Kinematics Value 8	X	X
T-0-0058	Kinematics Value 9	X	X
T-0-0059	Kinematics Value 10	X	X

Table 15-27: Coordinated Motion Parameters

Coordinated Motion Status

Parameter Number	Description	Valid for GPP 11	GMP 11
T-0-0100	Target Point Number	X	X
T-0-0101	Segment Status	X	X
T-0-0111	Current X Position	X	X
T-0-0112	Current Y Position	X	X
T-0-0113	Current Z Position	X	X

Table 15-28: Coordinated Motion Status Parameters

Task Status

Parameter Number	Description	Valid for GPP 11	GMP 11
T-0-0120	Task Operating Mode	X	X
T-0-0122	Task Diagnostic Message	X	X
T-0-0123	Task Status Message	X	X
T-0-0130	Current Instruction Pointer	X	X
T-0-0131	Current Instruction	X	X
T-0-0132	Instruction Pointer at Error	X	X
T-0-0133	Composite Instruction Pointer	X	X
T-0-0135	Current Subroutine	X	X
T-0-0136	Stack Variable Data	X	X
T-0-0137	Subroutine Breakpoint	X	X
T-0-0138	Sequencer Information	X	X
T-0-0200	Last Active Event Number	X	X

Table 15-29: Task Status Parameters

Task Parameter Lists

Parameter Number	Description	Valid for GPP 11	GMP 11
T-0-2000	List of All Parameters	X	X
T-0-2001	List of Required Parameters	X	X

Table 15-30: Task Parameters Lists

Axis Parameters - Class A

This section provides a list of all axis parameters that are available in the system. They are grouped by functionality and indicate the supported control firmware type.

Axis Setup Parameters

Parameter Number	Description	Valid for GPP 11	Valid for GMP 11
A-0-0001	Task Assignment	X	X
A-0-0002	Type of Positioning	X	X
A-0-0003	Axis Motion Type	X	X
A-0-0004	Axis Options	X	X
A-0-0005	Linear Position Units	X	X
A-0-0006	Reference Options	X	X
A-0-0007	Configuration Mode	X	X
A-0-0009	Drive PLS Register	X	X
A-0-0020	Maximum Velocity	X	X
A-0-0021	Maximum Acceleration	X	X
A-0-0022	Maximum Deceleration	X	X
A-0-0023	Jog Acceleration	X	X
A-0-0025	Maximum Jog Increment	X	X
A-0-0026	Maximum Jog Velocity	X	X

Table 15-31: Axis Setup Parameters

Axis Runtime Parameters

Parameter Number	Description	Valid for GPP 11	Valid for GMP 11
A-0-0100	Target Position	X	X
A-0-0101	Commanded Position	X	X
A-0-0102	Feedback Position	X	X
A-0-0103	Axis Modulo	X	X
A-0-0110	Programmed Velocity	X	X
A-0-0111	Commanded Velocity	X	X
A-0-0112	Feedback Velocity	X	X
A-0-0120	Programmed Acceleration	X	X
A-0-0121	Programmed Deceleration	X	X
A-0-0131	Sercos Control Word	X	X
A-0-0132	Sercos Status Word	X	X
A-0-0133	AT Error Count	X	X
A-0-0140	Mfg. Class 3 Status Word	X	X
A-0-0141	Torque Mode Commanded Torque	X	X
A-0-0142	Torque Feedback (cyclic)	X	X
A-0-0145	Current Motion Type	X	X

Table 15-32: Axis Status Parameters

Axis Electronic Line Shafting Parameters

Parameter Number	Description	Valid for GPP 11	Valid for GMP 11
A-0-0030	Ratio Mode Master Axis	X	X
A-0-0031	Control CAM/Ratio Master Factor (N)	X	X
A-0-0032	Control CAM/Ratio Slave Factor (M)	X	X
A-0-0033	Control CAM Stretch Factor (H)	X	X
A-0-0034	Control CAM Currently Active	X	X
A-0-0035	Control CAM Position Constant (L)	X	X
A-0-0036	Ratio Mode Encoder Type	X	X
A-0-0037	Ratio Mode Step Rate	X	X
A-0-0038	Ratio Mode Options	X	X
A-0-0150	Programmed Ratio Adjust	X	X
A-0-0151	Programmed Phase Offset	X	X
A-0-0153	Control Phase Adjust Average Velocity	X	X
A-0-0155	Control Phase Adjust Time Constant	X	X
A-0-0156	Phase Offset Velocity Feedback	X	X
A-0-0157	Current Phase/ Control CAM Master Offset	X	X
A-0-0158	Relative Phase Offset Distance Remaining	X	X
A-0-0159	Ratio Adjust Step Rate	X	X
A-0-0160	Commanded Ratio Adjust	X	X
A-0-0161	Control CAM Programmed Slave Adjust	X	X
A-0-0162	Control CAM Current Slave Adjust	X	X
A-0-0163	Control CAM Output Position	X	X
A-0-0164	ELS Options	X	X

Table 15-33: Axis Electronic Line Shafting Parameters

Axis Feedback Capture (Registration)

Parameter Number	Description	Valid for GPP 11	Valid for GMP 11
A-0-0170	Probe Configuration Status	X	X
A-0-0171	Probe 1 Positive Captured Value	X	X
A-0-0172	Probe 1 Negative Captured Value	X	X
A-0-0173	Probe 2 Positive Captured Value	X	X
A-0-0174	Probe 2 Negative Captured Value	X	X

Table 15-34: Axis Feedback Capture (Registration)

Optional Sercos Data

Parameter Number	Description	Valid for GPP 11	Valid for GMP 11
A-0-0180	Optional Command ID #1	X	X
A-0-0181	Optional Command ID #2	X	X
A-0-0182	Optional Command ID #3	X	X
A-0-0183	Optional Command ID #4	X	X
A-0-0184	Optional Command ID #5	X	X
A-0-0185	Optional Feedback ID #1	X	X
A-0-0186	Optional Feedback ID #2	X	X
A-0-0187	Optional Feedback ID #3	X	X
A-0-0188	Optional Feedback ID #4	X	X
A-0-0189	Optional Feedback ID #5	X	X
A-0-0190	Command Data #1	X	X
A-0-0191	Command Data #2	X	X
A-0-0192	Command Data #3	X	X
A-0-0193	Command Data #4	X	X
A-0-0194	Command Data #5	X	X
A-0-0195	Feedback Data #1	X	X
A-0-0196	Feedback Data #2	X	X
A-0-0197	Feedback Data #3	X	X
A-0-0198	Feedback Data #4	X	X
A-0-0199	Feedback Data #5	X	X

Table 15-35: Optional Sercos Data

Multiplexing Parameters (DKC 2.3 only)

Parameter Number	Description	Valid for GPP 11	Valid for GMP 11
A-0-0200	MDT Multiplex Selection List	X	X
A-0-0201	AT Multiplex Selection List	X	X
A-0-0202	MDT Multiplex Ident List	X	X
A-0-0203	AT Multiplex Ident List	X	X

Table 15-36: Multiplexing Parameters

Axis Parameter Lists

Parameter Number	Description	Valid for GPP 11	Valid for GMP 11
A-0-2000	List of All Parameters	X	X
A-0-2001	List of Required Parameters	X	X

Table 15-37: Axis Parameters Lists

15.6 Control Parameters – Class C

Control parameters include setup parameters for system configuration, program options, serial interface, and I/O options. Also included are status parameters such as operating mode, and diagnostic messages.

Note: The system parameters that appear in this section are **not** all valid for both GPP 11 and GMP 11 firmware. For this reason, any system parameter that is used specifically for GPP 11 or GMP 11 will be indicated at the end of the description.
For example:
"C-0-2640 PLC Connection Options ***(GMP only)***"
No specification to a firmware version indicates that the parameter is valid for both GPP 11 and GMP 11. Refer to the tables on page 15-5 for a complete listing of parameters by firmware type.

Control Parameter Activation Requirements

Some control parameters require a power cycle before a modified parameter value can take effect. Others, require a Sercos phase transition and/or write command to a parameters. The following tables detail which control parameters are affected and when a modified value takes effect.

Activation Requires a Power Cycle

The following control parameters require that the control's power be cycled before their modified values take effect:

Parameter	Description
C-0-0002	Control Address
C-0-0010	System options
C-0-0012	X16 Communication Port Device Type
C-0-0080	Maximum Number of Global Integers
C-0-0081	Maximum Number of Global Floats
C-0-0304	Link Ring Node Address
C-0-0305	Link Ring Cycle Time
C-0-0400	EtherNet IP Address
C-0-0401	EtherNet Subnet Mask
C-0-0402	EtherNet Gateway IP Address
C-0-0403	EtherNet Duplex Mode
C-0-1601	Integrated PLC: Configuration
C-0-2640	PLC Connection Options

Table 15-38: Control Parameters Requiring a Power Cycle

Activation after Entering and Exiting Sercos phase 2

The following control parameters take effect after entering and exiting Sercos phase 2 (Parameter mode):

Parameter	Description
C-0-0003	X10 Programming Port Setup
C-0-0004	X16 Communication Port Setup
C-0-0013	X10 Programming Port Mode
C-0-0014	X16 Communication Port Mode
C-0-0020	Transmitter Fiber optic Length

Table 15-39: Control Parameters Requiring a Sercos phase Transition

Activation on a Write to C-0-2601

The following control parameters take effect after writing to C-0-2601 in Sercos phase 2:

Parameter	Description
C-0-2600	Fieldbus Mapper: Cyclic Channel to PLC
C-0-2630	Fieldbus Slave Device Address
C-0-2631	Fieldbus Parameter / PCP Channel Length
C-0-2632	Fieldbus / PLC Multiplex Method
C-0-2633	Fieldbus Baud Rate (DeviceNet only)
C-0-2636	Fieldbus / PLC Word Swap

Table 15-40: Control Parameters Requiring a Write to C-0-2601

Activation after Entering and Exiting Sercos phase 2 or after a PLS Build and Activate Command

The following control parameters can take effect after entering and exiting Sercos phase 2 (Parameter mode) or after a PLS activate (C-0-2903) and PLS build (C-0-2905) command in Sercos phase 4:

Parameter	Description
C-0-2920	PLS1 Switch On List
C-0-2921	PLS1 Switch Off List
C-0-2922	PLS1 Switch Out List
C-0-2931	PLS1 Output Lead Time List
C-0-2932	PLS1 Output Lag Time List
C-0-2933	PLS1 Output One Shot List
C-0-2934	PLS1 Output Mode List
C-0-2935	PLS1 Output Direction List

Table 15-41: Control Parameters Requiring a Sercos phase Transition or PLS Build and Activate Commands

System Setup (C-0-0001 through C-0-0035)

C-0-0001 Language Selection

All parameter names, units and diagnostic warning messages within VisualMotion are available in English and German. This parameter is used to set the language for both the control and the drives. Although Rexroth digital drives supported multiple languages, VisualMotion will use English when a language other than German or English is set.

The following languages are supported:

		Supported by	
Value	Language	Control	Drive
0	Deutsch (German)	X	X
1	English	X	X
2	Francias (French)		X
3	Español (Spanish)		X
4	Itanlino (Italian)		X

Table 15-42: Supported Languages

Note: To change the language of a specific drive, use the Parameter Overview tool and modify Sercos parameter S-0-0265.

C-0-0001 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	4
Default value:	1
Access:	read / write in any Sercos phase

C-0-0002 Control Address

This parameter is used to set the control's address for the different interface scenarios listed in the table below. The default control address is 128.

Interface Scenario	Valid Range
Control Address	0-126 and 128 for SIS (127 is invalid)
RS-232 (RS-422) Serial Interface	0-126 and 128 for SIS (127 is invalid)
RS-485 Serial Interface	0-126 and 128 for SIS (127 is invalid)

Table 15-43: Control Address

The control address can be set in VisualMotion Toolkit by selecting ***Tools* ⇒ *Control Settings*** and changing the **Address** field under the *General* tab in offline or service mode.

Modifications to this parameter require that power to the control be cycled in order for the changes to take effect.

Control Address

Communication via Ethernet TCP/IP, Fieldbus (non-cyclic channel) and PCI/DPR (non-cyclic/programming channels) is supported for SIS protocols. The data received by the control must contain the same control address that is set in this parameter.

Serial Interface

RS-232, RS-422 and RS-485 serial interfaces are supported for SIS protocol. The data received by the control must contain the same address set in this parameter for both RS-232, RS-422 or RS-485 serial interface. Address 127 is reserved for a SIS master and cannot be used as a control address.

C-0-0002 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	128
Default value:	128
Access:	read / write in any Sercos phase
Activation:	requires power cycle

C-0-0003 X10 Programming Port Setup

This parameter sets the baud rate, checksum, and parity for serial port X10 on the control. The port can communicate with a PC, a PLC, HMI or any device that uses SIS protocol. The default settings of the port are 8 data bits, 1 stop bit, and no parity.

Checksum verification can be enabled by adding 1 to the baud rate (i.e. 9600 + 1 = 9601).

The parity can be changed by adding 0 for no parity, 4 for even parity, and 8 for odd parity to the baud rate/checksum value.

For example:

A port setting of 9600 baud rate with checksum enabled and even parity would equal 9605 (9600 + 1 + 4).

The supported baud rates are 9600, 19200, 38400, 57600 and 115200.

Modifications to this parameter require that power to the control be cycled in order for the changes to take effect.

C-0-0003 Attributes

Data length:	4 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	0
Maximum value:	115203
Default value:	9601 (baud rate + checksum on)
Access:	read / write in any Sercos phase
Activation:	activates after entering and exiting Sercos phase 2

C-0-0004 X16 Communication Port Setup

This parameter sets the baud rate, checksum, and status message for serial port X16 on the control. The port can communicate with a PC, a PLC, HMI or any device that uses SIS protocol. The default settings of the port are 8 data bits, 1 stop bit, and no parity. The device connected to this port is selected in parameter C-0-0012 X16 Communication Port Device Type.

Checksum verification and status message can be enabled by adding 1 for checksum and 2 for status message to the baud rate (i.e. 9600 + 1 + 2 = 9603). The supported baud rates are 9600, 19200, 38400, 57600 and 115200.

The parity can be changed by adding 0 for no parity, 4 for even parity, and 8 for odd parity to the baud rate/checksum/status message value.

For example:

A port setting of 9600 baud rate with checksum and status message enabled and even parity would equal 9607 (9600 + 1 + 2 + 4).

Modifications to this parameter require that power to the control be cycled in order for the changes to take effect.

C-0-0004 Attributes

Data length:	4 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	0
Maximum value:	115203
Default value:	115201 (baud rate + checksum on)
Access:	read / write in any Sercos phase
Activation:	activates after entering and exiting Sercos phase 2

C-0-0005 Communication Protocol Selection

This parameter is read-only and displays the current communication protocol recognized by VisualMotion for system communication. The allowable types are listed in the following table:

Value	Description
1	autosense for SIS protocol (set by VisualMotion)
2	reserved for future
3	reserved for future

Table 15-44: Communication Protocol Selection

C-0-0005 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	1
Maximum value:	3
Default value:	1
Access:	read-only

C-0-0008 System Control Word Status

This parameter is a status indicator for the System Control Word, C-0-0011. When the command is set, this parameter indicates the current status of the process.

The status of bits 1-4 are described in the following table:

Bit 4	Bit 3	Bit 2	Bit 1	Description
0	0	0	0	No Command
0	0	1	1	Command was successful
0	1	1	1	Command is processing
1	1	1	1	Error, command not successful

Table 15-45: System Control Word Status Bits 1-4

The status of the different command states are listed in the following section:

No Command

All bits (1-16) are set to 0.

Command was Successful

Bits 1 - 4 are set to 0b0011.

Bits 5 - 6 show the details of the success.

Bits 7 - 16 are set to 0.

Command is Processing

Bits 1 - 4 are set to 0b0111.

Bits 5 - 6 show the details of the success.

Bits 7 - 16 are set to 0.

Error, Command not Successful

Bits 1 - 4 are set to 0b1111.

Bits 5 - 6 show the details of the success.

Bits 7 - 16 is an error code that corresponds to the list of valid system diagnostic codes.

C-0-0008 Attributes

Data length:	2 byte data
Data type:	unsigned short
Display format:	binary
Units:	--
Minimum value:	0000 0000 0000 0000
Maximum value:	1111 1111 1111 1111
Default value:	0000 0000 0000 0000
Access:	read-only

C-0-0009 Error Reaction Mode

This parameter sets the reaction of the control to fatal errors as either immediate or controlled. When a fatal error occurs, the control determines which user task will be shutdown. Axes, associated with a user task that is shutdown, are stopped based on the error reaction of the drive. For detailed information on *Error Reaction*, refer to volume 1 of the *Rexroth VisualMotion 11 Functional Description*, *chapter 11*.

0 = Immediate error reaction to fatal errors

By default, all axes, regardless of their operating mode, are immediately disabled and stopped by the drive.

1 = Controlled error reaction to fatal errors

All motion comes to a controlled stop before the axes are disabled. ELS axes stop synchronized and coordinated motion axes stop on path. Axes in single axis mode and velocity mode are switched to drive halt.

C-0-0009 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	1
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-0010 System Options

This parameter sets several options for the VisualMotion system and for the Sercos ring.

```
0000000000000000
               |_ Bit 1

Bit    1: Simulation mode
Bits 2-3: Phase 2 Sercos IO Updates
Bit  4-5: Enable 2, 4, 8, 16 MBps Sercos Transmission
Bit    6: Generate Reco IO Errors
Bit    7: Ignore Reco IO Errors
Bit    8: Sercos Phase 2 Cycle Time
Bit    9: Not used
Bit   10: Reserved
Bit   11: Jog in auto mode
Bit   12: Prioritized service channel
Bit   13: Ignore drive warnings
Bit   14: Ignore axis ready status in program commands
Bit   16: Disable AT timing check
```

Fig. 15-2: Bit Description C-0-0010

C-0-0010 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	11111111 11111111
Default value:	000000000 1000000 (bit 7 set)
Access:	read in any Sercos phase / write in Sercos phase 2
Activation:	requires power cycle

Bit 1: Simulation Mode

When set to 1, the presence of axes will be simulated. The Sercos ring will not be scanned for drives, and the drive enable bits will be ignored. This mode is useful for simulating coordinated motion when the control is not connected to the actual system, or when a program does not contain any axes. All axis and task status parameters are simulated. Drive parameters and I/O are not simulated since they require a Sercos drive. Any drive-controlled motion (homing, single-axis, etc.) is also not simulated. ELS and single-axis modes are do not support this feature.

0 = normal drive operation

1 = simulation mode, do not scan for drives

Bits 2-3: Phase 2 Sercos I-O Update

In Sercos phase 2, all drive-resident I/O and Sercos I/O modules are scanned every 500ms by default (bits 2, 3 = 0). This can slow communications when downloading parameter lists from the drives, depending upon how much I/O is visible. If fast updates of I/O in Sercos phase 2 are not critical, set bits 2 and 3 to a nonzero value. The I/O will be scanned every 10 seconds or not at all, which speeds up communication with the user interface.

Bit 3	Bit 2	Update Time
0	0	500 ms update
0	1	10 second update
1	N/A	no update

Table 15-46: Phase 2 Sercos I-O Update

Bit 4-5: Enable 2, 4, 8, 16 MBps Sercos Transmission

Bits 4 and 5 sets the control's Sercos transmission rate to either 2, 4, 8, or 16 Mbits/s as follows:

Bit 5	Bit 4	Sercos Transmission Rate
0	0	2 Mbaud (default)
0	1	4 Mbaud
1	0	8 Mbaud
1	1	16 Mbaud

Table 15-47: Sercos Transmission Bit Settings

The transmission rate has no effect on the control or drive performance unless a smaller Sercos cycle time is needed. The minimum Sercos cycle time in control parameter C-0-0099 is 2ms.

Bit 6-7: RECO I/O Error Reaction Bits

Bits 6 and 7 are used to specify how the control responds to errors reported by RECO I/O modules. The bit settings are as follows:

Bit 6	Bit 7	Error
0	0	ignore
1	0	warning
x	1	fatal error (default)

Table 15-48: RECO I/O Error Reaction Bits (C-0-0010)

Ignore

The control ignores any errors reported by Local and Sercos RECO I/O modules. This reaction is selected if the user program is to handle RECO I/O errors. In this case, the I/O Setup tool can be used to map the RECO I/O modules to registers.

Refer to *volume 1* of the *Rexroth VisualMotion 11 Functional Description, section 7.2*.

Note: This reaction provides default backwards compatibility to older versions of GPP firmware that are not capable of directly responding to RECO I/O errors.

Warning

The control responds to errors reported by the Local and Sercos RECO I/O modules by issuing a "215 RECO I/O Failure" warning. This error reaction is selected if the user is to be notified of any RECO I/O errors, while still allowing the user program to continue executing.

Fatal Error

The control responds to errors reported by the Local and Sercos RECO I/O modules by issuing a "544 RECO I/O Failure" error, stopping program execution and motion. This is the system's default error reaction. This error reaction is selected if the application requires program execution and motion to be stopped as soon as a RECO I/O error is detected.

Bit 8: Sercos Phase 2 Cycle Time

When set to 0, the Sercos Cycle Time in phase 2 is set to 1 ms. This optimized time provides improved throughput in Sercos phase 2 communication.

When set to 1, the Sercos Cycle Time in phase 2 is set to 2 ms. This option provides a mechanism for support of certain Sercos devices that do not support an update rate of 1 ms in Sercos phase 2.

The setting in this parameter affects all axes in the system.

0 = 1 ms communication in Sercos phase 2 (default)

1 = 2 ms communication in Sercos phase 2

Note: Modifications to this parameter require that the system be rebooted for the changes to take effect.

Bit 11: Jog in Auto Mode

Any axis that is currently in single-axis, velocity, or coordinated motion mode can be jogged with the axis jog bits.

When set to 1, jogging can be performed in auto mode or when a task is running. This allows continuous or incremental axis motion to be started and stopped by simply setting its jog bits and parameters.

Bit 12: Prioritized Service Channel

Only one task or user interface can access a drive's Sercos service channel at a time. When drive parameter lists, long text strings, or oscilloscope data is transferred from one service channel access, other service channel access could be suspended for several seconds.

Set this bit to 1 if the timing for user tasks access, via the Sercos service channel, is critical. The user tasks will suspend any Sercos transmission of any text strings or lists from the user interface, and the "!78 Service channel in use" communication error will be issued.

If user tasks access is not critical, parameter lists and oscilloscope are seldom used during normal operation, or "!78" errors occur while viewing parameters, this bit should be set to 0.

0 = Don't prioritize the service channel (default)

1 = User tasks have priority over to user interface

Note: Even with prioritization, service channel access can vary between 10 to 100 ms. Therefore, drive parameters which transfer time critical data should be configured in the cyclic data channel. Refer to the *Sercos Drive Telegram Utility* on page 15-4 for details.

Bit 13: Ignore Drive Warnings

When set to 0, the control reacts to drive warnings by shutting down associated user tasks. A drive warning sets bit 13 (Class_2_Warning) in the relevant axis status register. The default reaction of the control to a warning is to issue the error "498 Drive Shutdown Warning".

When set to 1, drive warnings are ignored by the control and associated user tasks are not shutdown.

The setting in this parameter affects all axis in the system.

0 = Drive warnings issue shutdown (default)

1 = Drive warning are ignored by the control

Bit 14: Ignore Axis Ready Status in Program Commands

When set to 0, the control issues an error if a start command is instructed before the drive is ready (AH or AF). These commands include the start command, the homing command, and the operation mode switch. This is the control's default setting.

When set to 1, the control does not issue an error if the axis is not ready.

0 = Error if axis is not ready (default)

1 = Ignore error if axis is not ready

Bit 16: Disable AT Timing Check

When set to 0, the Sercos AT (Drive Telegram) time checking is enabled. This is the control's default setting.

When set to 1, Sercos AT time checking is disabled.

0 = Check Sercos AT timing (Default)

1 = Do not check AT timing

C-0-0011 System Control Word

This parameter is used by DriveTop to force the control into Sercos phase 2 or Sercos phase 4. In addition to forcing the system into Sercos phase 2, DriveTop uses a unique forcing of Sercos phase 4 in order to complete the safety technology configuration for IndraDrives. During the forcing of Sercos phase 4, no drive motion will be allowed ("Rf" or "Ah" signals are not set in the Sercos control word of the MDT). Register 1, bit 1 has no priority while the system is forced in Sercos phase 2 or 4. VisualMotion's archive and restore functionality also uses this parameter to force the system into Sercos phase 2.

Note: The current Sercos phase forcing state can be read in control parameter C-0-0120.

```
0000000000000000
               |_ Bit 1
Bit 1: Force Sercos Phase 2
Bit 2: Force Sercos Phase 4
```

Fig. 15-3: Bit Description C-0-0011

The following table shows the Sercos phase prioritization levels for the system:

VisualMotion Interface	Priority (A=highest)
C-0-0011, bit 1	A
C-0-0011, bit 2	B
Register 1, bit 1	C

Table 15-49: Sercos Phase Prioritization for the Control

During a Sercos phase forcing, one of the following system diagnostics will appear on the control's H1 display:

- 022 Sercos Phase 2 Forced
- 024 Sercos Phase 4 Forced

Bit 1: Force Sercos Phase 2

When this bit is set to a 1, the control is *forced* into Sercos phase 2 (parameter mode). The following behaviors exists:

- This bit overrides register 1 bit 1 as well as C-0-0011 bit 2.
- The PPC control will display status message “022 Sercos phase 2 Forced” when successfully implemented.
- Upon entering this state, if the system is already in Sercos phase 4 (manual or automatic mode or forced Sercos phase 4), then the system will be brought down to Sercos phase 0 and back up to P2. This is done because the “running state” of phase 4 is higher than the forced Sercos phase 2, and the mode handler needs to see the logical progression of low to high and then reset again.
- Upon entering this state, if the system is in parameter mode, then the system will NOT be brought down to Sercos phase 0 and then back up to P2. This is done because the “running state” of phase 2 is already the same as forced Sercos phase 2.
- Upon release of this bit, the system will diagnose the state of C-0-0011 bit 2 and register 1 bit 1 and set the Sercos phase accordingly.
- This bit is used by the VisualMotion archive utility to ensure the system stays in parameter mode regardless of the state of register 1 bit 1 (since it may be tied to IndraLogic which can’t be forced by VM). This bit is also used by DriveTop during the commissioning process of Safety Technology. When the drive server shuts down upon closing DriveTop, the bits in this parameter will be reset by the drive server interface to VisualMotion.
- Since the parameter is volatile, the bits are also reset upon power cycle of the control.

Bit 2: Force Sercos Phase 4 (drives disabled):

When this bit is set to a 1, the control is *forced* into Sercos phase 4. The following behaviors exists:

- This bit overrides register 1 bit 1 but can be overridden by C-0-0011 bit 1.
- The PPC control will show diagnostic “024 Sercos phase 4 Forced” when successfully implemented. The drives will be in Sercos phase 4 but will not have Rf or Ah signals set on the control. This implies that the drive state should be “Ab” or “bb” when successful. Note that the user tasks A – D can NOT be run in this state, but the IndraLogic tasks can run OK *without commanding motion*.
- Upon entering this state, if the system is already in Sercos phase 4 (manual or automatic mode), then the system will be brought down to Sercos phase 0 and back up to P4. This is done because the “running state” of phase 4 is higher than the forced Sercos phase 4, and the

mode handler needs to see the logical progression of low to high and then reset again.

- Upon entering this state, if the system is in parameter mode or Sercos phase 2 forced, then the system will NOT be brought down to Sercos phase 0 and then back up to P4. This is done because the “running state” of phase 2 is already lower than the forced Sercos phase 4.
- Upon exiting this state (release the bit), the system will diagnose the state of register 1 bit 1 and set the Sercos phase accordingly (keeping in mind the priority of C-0-0011 bit 1). Existing rules about Sercos phasing will apply (P0->P1->P2->P3->P4->P0->…).
- DriveTop uses this bit during the commissioning process of Safety Technology. When the drive server shuts down upon closing DriveTop, the bits in this parameter will be reset by the drive server interface to VisualMotion.
- Since the parameter is volatile, the bits are also reset upon power cycle of the control.

C-0-0011 Attributes

Data length:	2 byte data
Data type:	Binary number
Display format:	Binary
Units:	--
Minimum value:	0000000000000000
Maximum value:	11111111 11111111
Default value:	0000000000000000
Access:	read / write in any Sercos phase
Activation:	Immediate

C-0-0012 X16 Communication Port Device Type

This parameter selects the type of device that is connected to X16 on the control. Modifications to this parameter require that power to the control be cycled in order for the changes to take effect. The baud rate and options for these ports are configured in parameter C-0-0004 X16 Communication Port Setup. The allowable values are:

- 0 = off
- 1 = Standard Host (default)
- 2 = Teach Pendant
- 3 = Reserved
- 4 = IndraLogic
- 5 = SysLibCom

C-0-0012 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	5
Default value:	1
Access:	read in any Sercos phase / write in Sercos phase 2
Activation:	requires power cycle

C-0-0013 X10 Programming Port Mode

This parameter selects the communication mode for X10 on the control . This selection takes effect immediately. The baud rate and options for this port is configured in parameter C-0-0003 X10 Programming Port Setup. The allowable values are:

- 232 = RS232 (default)
- 422 = RS422
- 485 = RS485

C-0-0013 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	232
Maximum value:	485
Default value:	232
Access:	read / write in any Sercos phase
Activation:	activates after entering and exiting Sercos phase 2

RS-232

Connect to one device at a time according to RS-232 standard. Maximum cable length is 20 meters.

RS-422

Connect to one device at a time according to RS-422 standard. Maximum cable length is 200 meters.

RS-485

The control is a slave on a multi-drop ring with a host and up to 32 slaves, using the RS-485 standard. Maximum cable length is 200 meters.

C-0-0014 X16 Communication Port Mode

This parameter selects the communication mode for X16 on the control. This selection takes effect immediately. The baud rate and options for this port is configured in parameter C-0-0004 X16 Communication Port Setup.

C-0-0014 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	232
Maximum value:	485
Default value:	232
Access:	read / write in any Sercos phase
Activation:	activates after entering and exiting Sercos phase 2

C-0-0016 Communication Time-out Period

This parameter adjusts the communication time-out period. The state of the communication error timer is set to enabled/disabled by start/stop commands from the serial device. The communication timer is reset by both a Timer Reset command and a change of state from disabled to enabled.

C-0-0016 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	milliseconds
Minimum value:	50
Maximum value:	2000
Default value:	200
Access:	read / write in any Sercos phase

C-0-0017 X10 Programming Port Password

This parameter displays the current access level to the control over the X10 serial port. The available access levels are as follows:

C-0-0017	Access Level	Description
`007`	not defined (default)	no password is assigned and the user has full access to all control functions
`###`	Read/Write	password is assigned and the user has full access to all control functions
`***`	Read-only	password is assigned and the user only has read access

Table 15-50: X10 Programming Port Password

If this parameter displays "`###`" or "`***`", a password has been set and the user must enter the password in order to change the access level.

Note: Entering a string other than the correct password or cycling power to the control will change the control's current access level to "read-only".

Password Requirements:

The password can contain 3 to 10 alpha and/or numeric characters. The password is not case sensitive and special characters such as "$" or "%" are not allowed. Once set, the password is used to toggle between the different access levels.

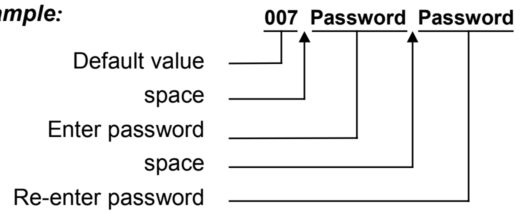
Every time the password is entered, the access level will toggle between "`###`" (Read/Write) and "`***`" (Read-only).

Note: The password for serial port X10 can only be set and accessed while connected to the port.

Setting a Password from the Default Value

When "`007`" is displayed in this parameter, use the following syntax to enter a password:

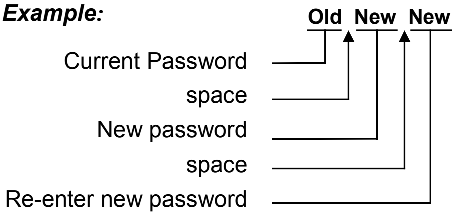
Example:



Changing an Existing Password

To change an existing password, enter the current password followed by the new password twice.

Example:



C-0-0017 Attributes

Data length:	10 character max
Data type:	unsigned character
Display format:	string
Units:	--
Minimum value:	0
Maximum value:	10
Default value:	007 (not defined)
Access:	read / write in Phase 2

C-0-0018 X16 Communication Port Password

This parameter displays the current access level to the control over the X16 serial port. Refer to *C-0-0017 X10 Programming Port Password* for details.

C-0-0020 Transmitter Fiber Optic Length

This parameter adjusts the intensity of the output from the control's Sercos transmitter, based on the length of the cable in meters. When using plastic fiber optic cable assemblies, set this parameter to match the length of the cable that is used between the control and the first drive's receiver (RX). For glass fiber, set this parameter to 50 m. Modifications to this parameter take effect when the control is switch in and out of parameter mode.

C-0-0020 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	meters
Minimum value:	+0.0
Maximum value:	+2000.0
Default value:	+0.0
Access:	read / write in any Sercos phase
Activation:	activates after entering and exiting Sercos phase 2

C-0-0021 User Watchdog Timer

The user watchdog timer enforces a time constraint on a user task specified in parameter C-0-0022.

Every time a nonzero timeout value is written to this parameter, a timer is triggered on the control. The task specified in parameter C-0-0022 must be completed within this time or error "*508 User Watchdog Timeout*" is issued. The timer is checked by the control every 50ms.

If a nonzero value is used, this parameter becomes active when the control is in Sercos phase 4, there are no errors, and the task specified in C-0-0022 is running.

C-0-0021 Attributes

Data length:	4 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	milliseconds
Minimum value:	0
Maximum value:	0
Default value:	0
Access:	read / write in any Sercos phase

C-0-0022 User Watchdog Task ID

This parameter selects the task that must be running in order for the watchdog timer in C-0-0021 to be active. If the watchdog function is not used, set parameter C-0-0021 to 0. The allowable values are:

0 = no task
1 = task A
2 = task B
3 = task C
4 = task D

C-0-0022 Attributes

Data length:	4 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	4
Default value:	0
Access:	read / write in any Sercos phase

Jogging and Display Parameters (C-0-0042 through C-0-0056)

C-0-0042 World Large Increment

This parameter is a scalar, for all tasks, that is multiplied with *T-0-0025 Maximum Jog Increment* to set the world large increment distance used when jogging a coordinated motion axis. The world large increment distance is used when bits 1 and 6 of the relevant task control register are set as follows:

Task Jog Control	Setting	Description
Registers 7-10	bit 1 = 0	nStep jogging (incremental)
	bit 6 = 1	large increment

Table 15-51: Register Setting for World Large Increment Jog Distance

For detailed information on *Registers 7- 10: Task Jog Control,* refer to *Chapter 16*.

C-0-0042 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	percent
Minimum value:	1
Maximum value:	100
Default value:	50
Access:	read / write in any Sercos phase

C-0-0043 World Small Increment

This parameter is a scalar, for all tasks, that is multiplied with *T-0-0025 Maximum Jog Increment* to set the world small increment distance used when jogging a coordinated motion axis. This parameter is used when bits 1 and 6 of the relevant task control register are set as follows:

Task Control	Setting	Description
Registers 7-10	bit 1 = 0	nStep jogging (incremental)
	bit 6 = 0	small increment

Table 15-52: Register Setting for World Small Increment Jog Distance

For detailed information on *Registers 7- 10: Task Jog Control,* refer to *chapter 16*.

C-0-0043 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	percent
Minimum value:	1
Maximum value:	100
Default value:	10
Access:	read / write in any Sercos phase

C-0-0045 World Fast Jog Speed

This parameter is a scalar, for all tasks, that is multiplied with *T-0-0026 Maximum Jog Velocity* to set the world fast jog speed when jogging a coordinated motion axis. This parameter is used when bits 1 and 6 of the relevant task control register are set as follows:

Task Control	Setting	Description
Registers 7-10	bit 1 = 1	continuous
	bit 6 = 1	fast jog speed

Table 15-53: Register Setting for World Fast Jog Speed

For detailed information on *Registers 7- 10: Task Jog Control,* refer to *chapter 16*.

C-0-0045 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	percent
Minimum value:	1
Maximum value:	100
Default value:	50
Access:	read / write in any Sercos phase

C-0-0046 World Slow Jog Speed

This parameter is a scalar, for all tasks, that is multiplied with *T-0-0026 Maximum Jog Velocity* to set the world slow jog speed when jogging a coordinated motion axis. This parameter is used when bits 1 and 6 of the relevant task control register are set as follows:

Task Control	Setting	Description
Registers 7-10	bit 1 = 1	continuous
	bit 6 = 0	slow jog speed

Table 15-54: Register Setting for World Slow Jog Speed

For detailed information on *Registers 7- 10: Task Jog Control,* refer to *chapter 16*.

C-0-0046 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	percent
Minimum value:	1
Maximum value:	100
Default value:	10
Access:	read / write in any Sercos phase

C-0-0052 Axis Large Increment

This parameter is a scalar that is multiplied with *A-0-0025 Maximum Jog Increment* to set the axis large increment distance used when jogging a an axis in single-axis mode. The axis large increment distance is used when bits 1 and 6 of the relevant task control register are set as follows:

Task Jog Control	Setting	Description
Registers 7-10	bit 1 = 0	nStep jogging (incremental)
	bit 6 = 1	large increment

Table 15-55: Register Setting for Axis Large Increment Jog Distance

Note: Axes in single-axis mode are jogged using their relevant axis control registers 11-18 and 209-240.

For detailed information on *Registers 7- 10: Task Jog Control* and *Registers 11-18: Axis Control* , refer to *chapter 16*.

C-0-0052 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	percent
Minimum value:	1
Maximum value:	100
Default value:	50
Access:	read / write in any Sercos phase

C-0-0053 Axis Small Increment

This parameter is a scalar that is multiplied with *A-0-0025 Maximum Jog Increment* to set the axis small increment distance used when jogging a an axis in single-axis mode. The axis small increment distance is used when bits 1 and 6 of the relevant task control register are set as follows:

Task Jog Control	Setting	Description
Registers 7-10	bit 1 = 0	nStep jogging (incremental)
	bit 6 = 0	small increment

Table 15-56: Register Setting for Axis Small Increment Jog Distance

Note: Axes in single-axis mode are jogged using their relevant axis control registers 11-18 and 209-240.

For detailed information on *Registers 7- 10: Task Jog Control* and *Registers 11-18: Axis Control* , refer to *chapter 16*.

C-0-0053 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	percent
Minimum value:	1
Maximum value:	100
Default value:	10
Access:	read / write in any Sercos phase

C-0-0055 Axis Fast Jog Velocity

This parameter is a scalar that is multiplied with *A-0-0026 Maximum Jog Velocity* to set the axis fast jog velocity used when jogging a an axis in single-axis mode. The axis fast jog velocity is used when bits 1 and 6 of the relevant task control register are set as follows:

Task Jog Control	Setting	Description
Registers 7-10	bit 1 = 1	continuous
	bit 6 = 1	fast jog velocity

Table 15-57: Register Setting for Axis Fast Jog Velocity

Note: Axes in single-axis mode are jogged using their relevant axis control registers 11-18 and 209-240.

For detailed information on *Registers 7- 10: Task Jog Control* and *Registers 11-18: Axis Control* , refer to *chapter 16*.

C-0-0055 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	percent
Minimum value:	1
Maximum value:	100
Default value:	50
Access:	read / write in any Sercos phase

C-0-0056 Axis Slow Jog Velocity

This parameter is a scalar that is multiplied with *A-0-0026 Maximum Jog Velocity* to set the axis slow jog velocity used when jogging a an axis in single-axis mode. The axis slow jog velocity is used when bits 1 and 6 of the relevant task control register are set as follows:

Task Jog Control	Setting	Description
Registers 7-10	bit 1 = 1	continuous
	bit 6 = 0	slow jog velocity

Table 15-58: Register Setting for Axis Slow Jog Velocity

Note: Axes in single-axis mode are jogged using their relevant axis control registers 11-18 and 209-240.

For detailed information on *Registers 7- 10: Task Jog Control, Registers 11-18: Axis Control and Registers 209-240: Axis Control,* refer to *chapter 16*.

C-0-0056 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	percent
Minimum value:	1
Maximum value:	100
Default value:	10
Access:	read / write in any Sercos phase

C-0-0070 Data Transfer Start of Program Floats

This parameter defines the start number for the range of program floats to be transferred between programs on the control. A value of zero identifies the smallest possible float number (default: float 1). This parameter works in conjunction with parameter C-0-0071 (the end number for the range of program floats). Together, these parameters created a range of program floats to be transferred.

An error is issued if the value this parameter is outside the range created by C-0-0070 and C-0-0071.

This value can be written directly into this parameter or defined in the *Data Transfer Ranges* tab by selecting ***Tools ⇒ Control Settings***.

C-0-0070 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	32767
Default value:	0
Access:	read / write in any Sercos phase

C-0-0071 Data Transfer End of Program Floats

This parameter defines the end number for the range of program floats to be transferred between programs on the control. A value of zero identifies the largest possible float number defined in the program. This parameter works in conjunction with parameter C-0-0070 (the start number for the range of program floats). Together, these parameters created a range of program floats to be transferred.

An error is issued if the value this parameter is outside the range created by C-0-0070 and C-0-0071.

This value can be written directly into this parameter or defined in the *Data Transfer Ranges* tab by selecting ***Tools ⇒ Control Settings***.

C-0-0071 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	32767
Default value:	0
Access:	read / write in any Sercos phase

C-0-0072 Data Transfer Start of Program Integers

This parameter defines the start number for the range of program integers to be transferred between programs on the control. A value of zero identifies the smallest possible integer number (default: integer 1). This parameter works in conjunction with parameter C-0-0073 (the end number for the range of program integers). Together, these parameters created a range of program integers to be transferred.

An error is issued if the value this parameter is outside the range created by C-0-0072 and C-0-0074.

This value can be written directly into this parameter or defined in the *Data Transfer Ranges* tab by selecting ***Tools ⇒ Control Settings***.

C-0-0072 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	32767
Default value:	0
Access:	read / write in any Sercos phase

C-0-0073 Data Transfer End of Program Integers

This parameter defines the end number for the range of program integers to be transferred between programs on the control. A value of zero identifies the largest possible integer number defined in the program. This parameter works in conjunction with parameter C-0-0072 (the start number for the range of program integers). Together, these parameters created a range of program integers to be transferred.

An error is issued if the value this parameter is outside the range created by C-0-0072 and C-0-0073.

This value can be written directly into this parameter or defined in the *Data Transfer Ranges* tab by selecting ***Tools ⇒ Control Settings***.

C-0-0073 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	32767
Default value:	0
Access:	read / write in any Sercos phase

C-0-0080 Maximum Number of Global Integers

This parameter defines the maximum number of global integers that will be created and accessible to the VisualMotion system. Modifications to this parameter require that power to the control be cycled in order for the changes to take effect. Global variables can be viewed using VisualMotion Toolkit by selecting ***Data ⇒ Variables…***.

C-0-0080 Attributes

Data length:	4 byte data
Data type:	unsigned long
Display format:	signed decimal
Units:	--
Minimum value:	512
Maximum value:	32767
Default value:	512
Access:	read / write in any Sercos phase
Activation:	requires power cycle

C-0-0081 Maximum Number of Global Floats

This parameter defines the maximum number of global floats that will be created and accessible to the VisualMotion system. Modifications to this parameter require that power to the control be cycled in order for the changes to take effect. Global variables can be viewed using VisualMotion Toolkit by selecting ***Data ⇒ Variables…***.

C-0-0081 Attributes

Data length:	4 byte data
Data type:	unsigned long
Display format:	signed decimal
Units:	--
Minimum value:	256
Maximum value:	32767
Default value:	256
Access:	read / write in any Sercos phase
Activation:	requires power cycle

C-0-0082 Save Global Variables Command

This parameter is used to save global integers and float variables to flash memory. When the control is powered, the values of all global variables are initialized to zero unless they were saved to flash memory before the control was powered down. The number of allowable global variables is determined by control parameters C-0-0080 (Global Integers) and C-0-0081 (Global Floats).

To save global variables…

1. Switch the control to parameter mode.
2. Edit C-0-0082 and set bits 1 and 2 to 1.
   C-0-0083, bits 1 and 2 display a 1 indicating a successful flash.
3. When completed, reset bits 1 and 2 to 0 and exit parameter mode.

Global variables are now stored in flash memory and can be reinitialized with saved values.

Note: Global variables can also be flashed using the Data Editor in VisualMotion Toolkit.

C-0-0082 Attributes

Data length:	2 byte data
Data type:	unsigned short
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00000011
Default value:	00000000 00000000
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-0083 Save Global Variables Status

This parameter is a status indicator for the *Save Global Variables Command parameter C-0-0082*. When the Save Global Variables Command is set, this parameter will indicate the communication status of the flash process. The status indications are described in the following table.

Bit 4	Bit 3	Bit 2	Bit 1	Description
0	0	0	0	No command to flash global variables
0	0	1	1	Command to flash global variables was successful
0	1	1	1	Command to flash global variables is processing.
1	1	1	1	Communication error, flash was not successful

Table 15-59: Save Global Variables Status Bits

C-0-0083 Attributes

Data length:	2 byte data
Data type:	unsigned short
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00001111
Default value:	00000000 00000000
Access:	read-only

Program Management (C-0-0090 through C-0-0099)

C-0-0090 Download Block Size

This parameter is used to set the block size that is used for user program downloads to the control.

C-0-0090 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	1
Maximum value:	110
Default value:	16
Access:	read / write in any Sercos phase

C-0-0091 Total Program Memory

This displays the total file memory on the control that can be used for user programs. This includes compiled programs and allocated points, variables, event and zone tables. The amount of total program memory varies based on the firmware version installed.

C-0-0091 Attributes

Data length:	4 byte data
Data type:	integer
Display format:	signed decimal
Units:	bytes
Minimum value:	--
Maximum value:	--
Default value:	total available program memory of installed firmware
Access:	read-only

C-0-0092 Available Program Memory

This parameter displays the amount of memory the control currently has available for storage of user programs. This does not include fragmented memory. Additional memory can be freed by clicking the **Defragment** button in the *Program Management* window.

C-0-0092 Attributes

Data length:	4 byte data
Data type:	integer
Display format:	signed decimal
Units:	bytes
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

C-0-0093 Contiguous Program Memory

This parameter displays the amount of memory the control currently has available for storage of user programs. This does not include fragmented memory. Additional memory can be freed by clicking the **Defragment** button in the *Program Management* window.

C-0-0093 Attributes

Data length:	4 byte data
Data type:	integer
Display format:	signed decimal
Units:	bytes
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

C-0-0094 Maximum Executable Program Size

VisualMotion reserves a fixed amount of memory to store the currently active executable program. The executable program contains the instructions but not the points, events, variables, and labels. This parameter indicates the largest executable program that can be stored on the control. The program size can be checked against this number before the program is downloaded. If the control receives a user program that exceeds this number, communication error “!60 Executable program is too large” is displayed.

C-0-0094 Attributes

Data length:	4 byte data
Data type:	integer
Display format:	signed decimal
Units:	bytes
Minimum value:	0
Maximum value:	0
Default value:	varies on firmware
Access:	read-only

C-0-0098 Initialization Delay

This parameter causes the control to delay for the specified number of seconds before it initializes the Sercos ring after power-up. This prevents the control from issuing a "No drives were found on ring" error if I/O stations or drives take a long time to initialize the Sercos ring.

C-0-0098 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	seconds
Minimum value:	0
Maximum value:	255
Default value:	4
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-0099 Sercos Cycle Time

This parameter sets the Sercos cycle time used by the control. During system initialization, the control uses this time as a starting value for the internal Sercos cycle time layer. For applications that require more calculating time, such as with high axis counts, lower Sercos baud rates, DAQ cards, or complex control motion such as coordinated motion, coordinated articulation or CAM Indexer, the control will perform an internal check and determine if this value is sufficient. If the time in this parameter is too small for performing the necessary calculations, the control will automatically change the internal Sercos cycle time and modify C-0-0099, all drive's Sercos cycle time parameter S-0-0002 and NC cycle time parameter S-0-0001 to match. If the value in this parameter is greater then the internal check performed by the control, the value in C-0-0099 will be used by the control and drives.

C-0-0099 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	microseconds
Minimum value:	2000
Maximum value:	16000
Default value:	2000
Access:	read in any Sercos phase / write in Sercos phase 2

Note: If larger Sercos cycle times are used and communication errors continue, increase the VisualMotion Dde Server's *Response Timeout* under ***Setting ⇒ Server Configuration*** menu selection.

System Status (C-0-0100 through C-0-0142)

System status messages are available through the communication ports and the user program, and provide the current status of the VisualMotion system.

C-0-0100 Control Firmware Version

This parameter displays the firmware version number issued by the control.

Format:

```
PFM01*-GPP-11V01-MS
 |       |     |_______ Firmware version
 |       |_____________ Firmware type (GPP or GMP)
 |_____________________ Hardware type
```

C-0-0100 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	installed firmware version
Access:	read-only

C-0-0101 Control Hardware Version

This parameter displays the control's hardware version.

Format:

```
PPC-R22.102x.1N-T-NN-NN-NN-FW Rev:3.0
|      |      |_____|      |_ Board revision
|      |         |___ Configuration
|      |_____________ Hardware revision no.
|____________________ Hardware platform
```

C-0-0101 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	installed hardware configuration
Access:	read-only

C-0-0102 Control Version Date

This parameter displays the release date for the firmware detected by the control.

C-0-0102 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	installed firmware release date
Access:	read-only

C-0-0103 Allowable Drive Address Range

This parameter displays the maximum Sercos drive address that can be set in any Sercos device within the control's fiber optic ring. The Sercos drive address is set using the S2 and S3 rotary selector switches.

For detailed information on Sercos drive address, refer to the *VisualMotion 11 Project Planning* manual.

C-0-0103 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	1
Maximum value:	99
Default value:	4
Access:	read-only

C-0-0104 Bootloader Firmware Version

This parameter displays the firmware version of the installed Bootloader in the PFM memory card.

C-0-0104 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	installed Bootloader firmware
Access:	read-only

C-0-0120 Operating Mode

This parameter displays the control's current mode of operation. The allowable values are:

Operating Mode	Value
Sercos phase 4 Forced	-2
Reserved	-1
Initializing control	0
Parameter Mode / Sercos phase 2 Forced	1
Manual / Automatic / Run Mode	2
Reserved	3

Table 15-60: Operating Mode States

C-0-0120 Attributes

Data length:	1 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	-2
Maximum value:	3
Default value:	0
Access:	read-only

C-0-0121 Sercos Communication Phase

This parameter displays the current Sercos initializing Sercos phase of the control.

Note: The drives can be at a lower Sercos phase than the control if an error exists at the drive level.

C-0-0121 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	4
Default value:	current Sercos phase
Access:	read-only

C-0-0122 Diagnostic Message

This parameter displays the current system status or error message issued by the control. During initialization, a **Msg1** icon in the initialization task sets this message.

For example:

400 Emergency Stop

C-0-0122 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	Highest priority message
Access:	read-only

For a complete listing of all system status and error codes, refer to the *Rexroth VisualMotion 11 Troubleshooting Guide*.

C-0-0123 Diagnostic Code

This parameter displays the current system status or error message number issued by the control.

For example:

4 for 004 Emergency Stop

C-0-0123 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	unsigned decimal
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	Highest priority code
Access:	read-only

For a complete listing of all system status and error codes, refer to the *Rexroth VisualMotion 11 Troubleshooting Guide*.

C-0-0124 Extended Diagnostic

This is a dynamic system message used to provide additional diagnostic information for a status warning or error message C-0-0122 Diagnostic Message.

C-0-0124 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	dependent on C-0-0122
Access:	read-only

For a complete listing of all system status and error codes, refer to the *Rexroth VisualMotion 11 Troubleshooting Guide*.

C-0-0125 System Timer Value

This general-purpose timer continuously counts in milliseconds while the control is running. It can be read into an integer variable via the **Param1** icon to provide timing for a section of a user program, or its incremental value can be used to time a process. It is a 31 bit counter with a maximum count of 2,147,483,647 (2^31), after which it rolls over to 0 and continues counting. It can be set to any value by the user program or the user interface.

C-0-0125 Attributes

Data length:	4 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	milliseconds
Minimum value:	0
Maximum value:	2^{31}
Default value:	increments by 1 every msec
Access:	read / write in any Sercos phase

C-0-0126 Date and Time

This parameter contains the date and time used by the control for the diagnostic log. VisualMotion controls do **not** contain a real-time clock. For this reason, the host device must set the date and time after the control is powered up.

When using VisualMotion Toolkit, the date and time can be set by clicking on the *Set Time* button under menu selection ***Diagnostics* ⇒ *System* ⇒ *Diagnostic Log*** or by directly modifying this parameter.

Format:

```
Month-Day-Year    Hour-Minute-Second in 24 hour format
06-22-2000        15:33:00
```

C-0-0126 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	01-01-1980 00:00:00 (at power up)
Access:	read / write in any Sercos phase

C-0-0127 Current Control Temperature (GPP only)

This parameter displays the current internal temperature of the control hardware in degrees Celsius.

Note: The maximum allowable internal temperature for the control is 70°C (158°F) when operating at a maximum ambient temperature of 45°C (113°F).

C-0-0127 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	degrees
Minimum value:	--
Maximum value:	--
Default value:	current operating temperature
Access:	read-only

C-0-0128 Elapsed Time Operational Counter

This parameter displays the value of the Elapsed Time Counter which continuously counts the number of system operation hours. The value in this parameter is updated 10,000 times every hour. Elapsed Time Counter values are stored in the control's EEPROM every 10 minutes so that the counter is not disrupted when the control is powered off or when firmware is changed, if the new firmware is the same or a later version. Versions of firmware earlier than GPP 11 do not contain this parameter.

C-0-0128 Attributes

Data length:	4 byte data
Data type:	float
Display format:	signed decimal
Units:	hours
Minimum value:	0.00
Maximum value:	1,080,000.00
Default value:	0.00
Access:	read-only

C-0-0142 Control Name

An alpha-numeric descriptive name, up to a maximum of 80 characters, assigned to the control is stored in this parameter (e.g., Master PPC). It has no functional significance.

C-0-0142 Attributes

Data length:	variable 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	<undefined>
Access:	read / write in any Sercos phase

Save CAM Built to Flash (C-0-0166 through C-0-0168)

C-0-0166 Save Built CAM to Flash ID

This parameter specify a valid id number, between 1 and 37, for a built control CAM that will be flashed when C-0-0167 Save Built CAM to Flash Command is used.

C-0-0166 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	--
Units:	--
Minimum value:	0
Maximum value:	37
Default value:	0
Access:	read / write in any Sercos phase

C-0-0167 Save Built CAM to Flash Command

This parameter sets the command to flash the built control CAM specified in Save CAM Built to Flash (C-0-0166 through C-0-0168)

C-0-0166 Save Built CAM to Flash ID.

To save a built control CAM…

1. Switch the control to parameter mode.
2. Edit C-0-0167 and set bits 1 and 2 to 1.
   C-0-0168, bits 1 and 2 display a 1 indicating a successful flash.
3. When completed, reset bits 1 and 2 to 0 and exit parameter mode.

Note: The **Param1** icon can be used to execute and reset this command.

C-0-0167 Attributes

Data length:	2 byte data
Data type:	unsigned short
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00000011
Default value:	00000000 00000000
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-0168 Save Built CAM to Flash Status

This parameter is a status indicator for the *Save Built CAM to Flash Command C-0-0167*. When the command is set, this parameter will indicate the status of the flash process and the ID number of the control CAM that was flashed. The status indications are described in the following table.

Bit 4	Bit 3	Bit 2	Bit 1	Description
0	0	0	0	No command to flash CAM
0	0	1	1	Command to flash CAM was successful and CAM ID has been encoded in the upper-byte
0	1	1	1	Command to flash CAM is processing
1	1	1	1	Communication error, flash was not successful

Table 15-61: Lower-byte Flash Status

The upper-byte (bits 9-16) of this parameter will display the CAM's ID number upon a successful flash. If the command to flash fails, the lower-byte will display a binary 15 and an error code will be displayed in the upper-byte. The following table lists the error codes that can occur:

Error Code	Description
23	"Insufficient Program Space" There is not enough free memory in the flash file system to save the specified CAM.
74	"Error in command execution" A general error occurred during the control’s attempt to save the specified CAM to flash.
104	"Invalid CAM ID" The CAM ID written to C-0-0166 must be between 1 and 37.
105	"CAM Does Not Exist" The CAM specified in C-0-0166 does not exist, it has not been built or downloaded.
106	"CAM is Not Ready" The CAM specified in C-0-0166 is not in a ready state. Its build process may not have yet completed.
107	"CAM is In Use" The CAM specified in C-0-0166 is currently in use.

Table 15-62: Upper-byte Flash Status

C-0-0168 Attributes

Data length:	2 byte data
Data type:	unsigned short
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	--
Default value:	00000000 00000000
Access:	read-only

NVRam and Compact Flash (C-0-0170 through C-0-0175)

C-0-0170 NVRam Backup Command

This parameter sets the command to copy the contents of the NVRam(non-volatile memory) to the compact flash memory card.

To copy the NVRam…

1. Switch the control to parameter mode.
2. Edit C-0-0170 and set bits 1 and 2 to 1.
3. When completed, C-0-0171, bits 1 and 2 are set to 1.
   Reset C-0-0170, bits 1 and 2 to 0 and exit parameter mode.

Note: The **Param1** icon can be used to execute and reset this command.

The NVRam on VisualMotion controls store the file system, parameters, and variables. On power up, the contents of the NVRam is automatically copied to compact flash. If the control is to be replaced, the validity of the NVRam should be checked. Register 21, bit 16 is set to 1 when the contents of the NVRam is valid. If the bit is set to 0, perform this command before moving the compact flash to a different control.

C-0-0170 Attributes

Data length:	2 byte data
Data type:	unsigned short
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00000011
Default value:	00000000 00000000
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-0171 NVRam Backup Status

This parameter is a status indicator for the *Store Parameters to Flash Command C-0-0170*. When the command is set, this parameter will indicate the status of the flash process. The status indications are described in the following table.

Bit 4	Bit 3	Bit 2	Bit 1	Description
0	0	0	0	No command to flash
0	0	1	1	Command to flash was successful
0	1	1	1	Command to flash is processing
1	1	1	1	Execution error, flash was not successful

Table 15-63: Store Parameters to Flash Status

C-0-0171 Attributes

Data length:	2 byte data
Data type:	unsigned short
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00001111
Default value:	00000000 00000000
Access:	read-only

C-0-0172 Compact Flash Backup Command

This parameter sets the command to perform a complete backup of the compact flash. This command can only be set for compact flash memory cards that are 32MB or larger. The memory area of the compact flash card is divided in half. The lower half contains the firmware and file system. The upper half is reserved for a backup copy.

To backup the compact flash…

1. Switch the control to parameter mode.
2. Edit C-0-0172 and set bits 1 and 2 to 1.
3. When completed, C-0-0173, bits 1 and 2 are set to 1.
   Reset C-0-0173, bits 1 and 2 to 0 and exit parameter mode.

Note: This command can only be set by accessing the parameter directly and not through the user program.

C-0-0172 Attributes

Data length:	2 byte data
Data type:	unsigned short
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00000011
Default value:	00000000 00000000
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-0173 Compact Flash Backup Status

This parameter is a status indicator for the *Backup compact flash - Command C-0-0172*. When the command is set, this parameter will indicate the status of the backup process. The status indications are described in the following table.

Bit 4	Bit 3	Bit 2	Bit 1	Description
0	0	0	0	No command to backup compact flash
0	0	1	1	Command to backup was successful
0	1	1	1	Command to backup is processing
1	1	1	1	Execution error, backup was not successful

Table 15-64: Backup compact flash - Status

C-0-0173 Attributes

Data length:	2 byte data
Data type:	unsigned short
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00001111
Default value:	00000000 00000000
Access:	read-only

C-0-0174 Compact Flash Restore Command

This parameter sets the command to perform a restore of the compact flash. This command can only be set for compact flash memory cards that are 32MB or larger containing a valid backup. Once the restore command is performed, the control is automatically rebooted.

To restore the compact flash…

1. Switch the control to parameter mode.
2. Edit C-0-0174 and set bits 1 and 2 to 1.
3. When completed, C-0-0175 bits 1 and 2 are set to 1.
   Reset C-0-0174 bits 1 and 2 to 0 and exit parameter mode.

Note: This command can only be set by accessing the parameter directly and not through the user program.

C-0-0174 Attributes

Data length:	2 byte data
Data type:	unsigned short
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00000011
Default value:	00000000 00000000
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-0175 Compact Flash Restore Status

This parameter is a status indicator for the *Restore compact flash - Command C-0-0175*. When the command is set, this parameter will indicate the status of the backup process.. The status indications are described in the following table.

Bit 4	Bit 3	Bit 2	Bit 1	Description
0	0	0	0	No command to restore compact flash
0	0	1	1	Command to restore was successful
0	1	1	1	Command to restore is processing
1	1	1	1	Execution error, restore was not successful

Table 15-65: Restore compact flash - Status

C-0-0175 Attributes

Data length:	2 byte data
Data type:	unsigned short
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00001111
Default value:	00000000 00000000
Access:	read-only

C-0-0176 Compact Flash: Date / Time of Backup

This parameter stores the date and time of the last commanded compact flash backup. The compact flash backup command is executed using control parameter C-0-0172 or by selecting ***Commission ⇒ Archive ⇒ Other…*** in VisualMotion Toolkit while online with the project.

Format:

```
Month-Day-Year    Hour-Minute-Second in 24 hour format
06-22-2000        15:33:00
```

C-0-0176 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	0
Maximum value:	20
Default value:	last commanded compact flash backup date and time
Access:	read / write in any Sercos phase

C-0-0177 Compact Flash: Firmware Version of Backup

This parameter displays the stored firmware version of the last compact flash backup command.

Format:

```
PFM01*-GPP-11V01-MS
 |      |    |_______ Firmware version
 |      |____________ Firmware type (GPP or GMP)
 |___________________ Hardware type
```

C-0-0177 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	0
Maximum value:	40 characters
Default value:	last firmware version backed up
Access:	read-only

Control Processor Usage Status (C-0-0200 through C-0-0203)

C-0-0200 Current Load due to Motion

This parameter displays the current amount of time required by the control's processor to process high priority motion task as a percentage of parameter C-0-0099(Sercos Cycle Time).

For example:

If C-0-0200 = 10% and C-0-0099 = 2 ms, then the amount of time to process high priority motion task is 200 µs.

C-0-0200 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	percent
Minimum value:	+0.0
Maximum value:	--
Default value:	current processing time is displayed
Access:	read-only

C-0-0201 Peak Load due to Motion

This parameter displays the peak amount of time encountered by the control for processing high priority motion task as a percentage of parameter C-0-0099 (Sercos Cycle Time). This parameter is written to by the control during Phase 4. This parameter is reset at power-down or when switching to Phase 2.

C-0-0201 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	percent
Minimum value:	+0.0
Maximum value:	--
Default value:	last peak processing time encountered
Access:	read-only

C-0-0202 Current Load due to I/O

This parameter displays the current amount of time required by the control's processor to process all configured Inputs and Outputs as a percentage of parameter C-0-0099 (Sercos scan time).

C-0-0202 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	percent
Minimum value:	+0.0
Maximum value:	+100.0
Default value:	current processing time is displayed
Access:	read-only

C-0-0203 Peak Load due to I/O

This parameter displays the peak amount of time encountered by the control for processing all configured Inputs and Outputs.

C-0-0203 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	percent
Minimum value:	+0.0
Maximum value:	--
Default value:	last peak processing time encountered
Access:	read-only

Link Ring Parameters (C-0-0300 through C-0-0303)

C-0-0300 Link Ring Control Word

This parameter is used for configuring a control as a Link Ring participant. The type of ring structure is also set in the control word.

```
0000000000x00xx
          |  ||  Control link via DAQ card
          |  ||  Bit 1 = 0 (passive)
          |  |   Bit 1 = 1 (active)
          |  |
          |  |   Control Function
          |  |   Bit 2 = 0 (Link slave)
          |      Bit 2 = 1 (Link master)
          |
          |      Ring Structure
                 Bit 5 = 0 (Single ring)
                 Bit 5 = 1 (Double ring)
```

Fig. 15-4: Bit Description C-0-0300

The corresponding bits are listed in the tables below.

C-0-0300 Bit 2	C-0-0300 Bit 1	Control behavior in the link	Control function in the link
1	1	Active participant	Link master
0	1	Active participant	Link slave
x	0	Passive participant	repeater

C-0-0300 Bit 5	Ring structure
0	Single ring
1	Double ring

Table 15-66: Link Ring Control Word Bits

C-0-0300 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0000000000000000
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-0301 Link Ring Primary Fiber Optic Length

This parameter is used to adjust the output power of the DAQ card to the length of the connected primary fiber optic cable. The length indicated relates to the fiber optic cable from X52 (TX) to the next connected DAQ, X53 (RX). A value less than 1.1 will configure the DAQ for a fiber optic cable up to 1.1 meters long.

C-0-0301 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	meters
Minimum value:	+0.0
Maximum value:	+2000.0
Default value:	+0.0
Access:	read / write in any Sercos phase

C-0-0302 Link Ring Secondary Fiber Optic Length

This parameter is used to adjust the output power of the DAQ card to the length of the connected secondary fiber optic cable. The length indicated relates to the fiber optic cable from X70 (TX) to the next connected DAQ, X71 (RX). A value less than 1.1 will configure the DAQ for a fiber optic cable up to 1.1 meters long.

C-0-0302 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	meters
Minimum value:	+0.0
Maximum value:	+2000.0
Default value:	0.0
Access:	read / write in any Sercos phase

C-0-0303 Link Ring MDT Error Counter

Every Link Ring slave counts the number of invalid master data telegrams (MDT). The control will issue error "541 Link Ring Error" and react with the parameterized error reaction if more than one MDT is invalid. The error counter stops if transmission is severely disrupted and the value 65535 will be displayed after an extended period of time. This parameter is reset on power-up.

C-0-0303 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	0
Default value:	0
Access:	read-only

C-0-0304 Link Ring Node Address

Up to 32 controls can be used in a Link Ring configuration. Link Ring does not use SIS protocol for communication. This parameter sets the address for each control used in a Link Ring configuration. Each control must have its own unique device address.

The valid range is 1 to 32.

Modifications to this parameter require that power to the control be cycled in order for the changes to take effect.

C-0-0304 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	1
Maximum value:	32
Default value:	1
Access:	read / write in any Sercos phase
Activation:	requires power cycle

C-0-0305 Link Ring Cycle Time

This parameter is used to set the Link Ring cycle time in microseconds. The cycle time is based on the maximum number of Link Ring nodes and their respective minimum Sercos cycle time (C-0-0099) setting. During system initialization, the Link Ring master uses this time to configure the communication between each Link Ring node.

Refer to the following table when setting the Link Ring cycle time:

Maximum Number of Link Ring Nodes	Required Addressing and Ordering	Link Ring Cycle Time
8	1 – 8	2 ms
16	1 – 16	4 ms
32	1 – 32	8 ms

Table 15-67: Link Ring Cycle Time Setting

Note: The minimum Sercos cycle time (C-0-0099) for each participating Link Ring node is automatically adjusted to the corresponding value set in C-0-0305.

Modifications to this parameter require that power to the control be cycled in order for the changes to take effect.

C-0-0305 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	unsigned decimal
Units:	microseconds
Minimum value:	2000
Maximum value:	8000
Default value:	8000
Access:	read in any Sercos phase / write in Sercos phase 2
Activation:	requires power cycle

Ethernet Parameters (C-0-0400 through C-0-0420)

The PPC-R22.1 control hardware, using GPP11 firmware, can simultaneously support an onboard Ethernet interface and the 10/100 Ethernet card. The onboard EtherNet and EtherNet option card both support standard TCP/IP and EtherNet/IP communication. When both Ethernet interfaces are installed, the Ethernet option card uses control parameter set C-0-0400 through C-0-0409. The onboard Ethernet interface will then use set C-0-0411 through C-0-0420. If only one interface is installed, then only C-0-0400 through C-0-0409 will be used.

C-0-0400 Ethernet 1 IP Address (GPP only)
C-0-0411 Ethernet 2 IP Address

This parameter contains a unique IP Address, specified in dot notation i.e., 172.16.11.200, assigned to the optional Ethernet interface on the PPC-R22.1. This parameter is valid for standard Ethernet TCP/IP and Ethernet/IP communication using 10/100 Mbps auto-negotiation. The IP address in this parameter must match the IP address configured in the communication server. A default value of 192.168.0.2 is defined to allow for a direct connect to the control from a PC using a crossover Ethernet cable (EtherNet to EtherNet)

Modifications to this parameter require that power to the control be cycled in order for the changes to take effect.

Note: The assignment of a unique IP address is the responsibility of the user's Information and Technology department.

C-0-0400 & C-0-0411 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	192.168.0.2
Access:	read / write in any Sercos phase
Activation:	requires power cycle

C-0-0401 Ethernet 1 Subnet Mask (GPP only)
C-0-0412 Ethernet 2 Subnet Mask

This parameter contains a subnet mask, specified in dot notation, used to determine what subnet is assigned to the IP address in parameter C-0-0400. This parameter is valid for standard Ethernet TCP/IP and Ethernet/IP communication using 10/100 Mbps auto-negotiation.

Modifications to this parameter require that power to the control be cycled in order for the changes to take effect.

C-0-0401 & C-0-0412 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	255.255.255.0
Access:	read / write in any Sercos phase
Activation:	requires power cycle

C-0-0402 Ethernet 1 Gateway IP Address (GPP only) C-0-0413 Ethernet 2 Gateway IP Address

This parameter contains the Gateway IP Address, specified in dot notation i.e., 172.16.1.1, assigned to the optional Ethernet card. This parameter is valid for standard Ethernet TCP/IP and Ethernet/IP communication using 10/100 Mbps auto-negotiation. A Gateway IP address identifies the router to which the IP Address is assigned. This parameter can be configured in VisualMotion Toolkit by selecting ***Tools* ⇒ *Control Settings*** and selecting the **Ethernet** tab.

Modifications to this parameter require that power to the control be cycled in order for the changes to take effect.

C-0-0402 & C-0-0413 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	192.168.0.1
Access:	read / write in any Sercos phase
Activation:	requires power cycle

C-0-0403 Ethernet 1 Duplex Mode (GPP only)

This parameter specifies whether the control is connected to a half or full duplex switch. To change the duplexing mode for network communication, the user must enter the following text using all caps.

- HALF
- FULL

Note: This parameter is valid for standard Ethernet TCP/IP and Ethernet/IP communication when using the ETH01 10 Mbps interface card. The ETH02 10/100 Mbps interface card and onboard Ethernet interface support auto-negotiation and do not require a duplex mode setting.

C-0-0403 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	Half-duplex Mode
Access:	read / write in any Sercos phase
Activation:	requires power cycle

C-0-0404 Ethernet 1 Access Control (GPP only)
C-0-0415 Ethernet 2 Access Control

This parameter is used to enable the control's access level (Bit 1) and to enable the optional Ethernet interface as either standard Ethernet TCP/IP or Ethernet/IP (Bit 2). This parameter is valid for standard Ethernet TCP/IP and Ethernet/IP communication using 10/100 Mbps auto-negotiation.

Modifications to this parameter require that power to the control be cycled in order for the changes to take effect.

Network Access Level (Bit 1)

Bit 1 of this parameter works in conjunction with C-0-0405 EtherNet 1 Interface Password to enable the control's network access level. The available access levels are Read-only, Read/Write or No Access.

Note: Before enabling the control's access level in this parameter, a password should be set in C-0-0405. If no password is set, the network access level can not be set to "No Access". The user will simply access the control in either the Read/Write or Read-only levels (Determined by C-0-0405).

C-0-0405 password levels:
`007` = Not defined (default)
`###` = Read/Write
`***` = Read-only

The following table shows the combinations that are required for both control parameters C-0-0404 and C-0-0405.

C-0-0404 (Bit 1)	C-0-0405	Access Level	Description
000000000000000**x**	007	not defined (default)	user has full access to all control functions
000000000000000**0**	***	Read-only	only read access has been allowed
000000000000000**x**	###	Read/Write	the user has full access to all control functions
000000000000000**1**	***	No Access	the control can not be access via Ethernet unless the correct password is entered in C-0-0405

Table 15-68: Enable Network Access Level C-0-0404, Bit 1

Once C-0-0404, bit 1 is set to 1, the network access level can be toggle, using a serial or Ethernet connection, between "***" (No Access) or "###" (Read/Write) by entering the correct password in C-0-0405.

Note: Entering a string other than the correct password or cycling power to the control will change the control's network access to "No Access". Entering the password again will change the control's network access back to "Read/Write".

Enable Ethernet/IP (Bit 2)

Bit 2 of this parameters enables the optional Ethernet card as either standard Ethernet TCP/IP or Ethernet/IP communication.

C-0-0404 (Bit 2)	Description
0000000000000000	standard Ethernet TCP/IP enabled (default)
0000000000000010	Ethernet/IP enabled

Table 15-69: Enable Ethernet C-0-0404, Bit 2

C-0-0404 & C-0-0415 Attributes

Data length:	2 byte data
Data type:	unsigned short
Display format:	binary
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	00000000 00000000
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-0405 EtherNet 1 Interface Password (GPP only)
C-0-0416 EtherNet 2 Interface Password

This parameter displays the current access level to the control over an EtherNet connection when using the VisualMotion Dde server. The available access levels are as follows:

C-0-0405	Access Level	Description
007	not defined (default)	no password is assigned and the user has full access to all control functions
###	Read/Write	password is assigned and the user has full access to all control functions
***	Read-only	password is assigned and the user only has read access

Table 15-70: EtherNet Interface Password

If this parameter displays "###" or "***", a password has been set and the user must enter the password in order to change the access level.

Note: Entering a string other than the correct password or cycling power to the control will change the control's access level to "Read-only".

Password Requirements:

The password can contain 3 to 10 alpha and/or numeric characteristics. The password is not case sensitive and special characters such as "$" or "%" are not allowed. Once set, the password is used to toggle between the different network access levels.

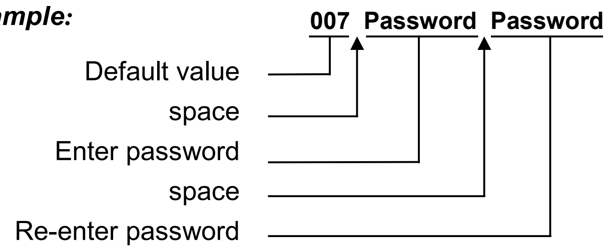
Every time the password is entered, the access level will toggle between "###" (Read/Write) and "***" (Read-only).

Note: If parameter C-0-0404 is set to 1 and C-0-0405 is set to "Read-only", the control will be set to "No Access" for all users connected to the control via a network connection. Refer to *C-0-0404 Ethernet 1 Access* Control for details.

Setting a Password from the Default Value

When "007" is displayed in this parameter, use the following syntax to enter a password:

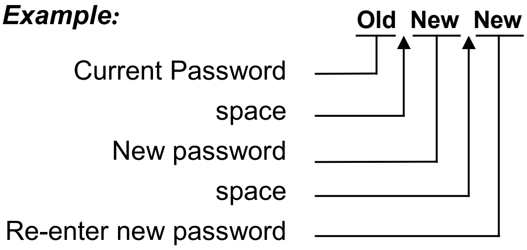
Example:



Changing an Existing Password

To change an existing password, enter the current password followed by the new password twice.

Example:



C-0-0405 & C-0-0416 Attributes

Data length:	10 character max
Data type:	unsigned character
Display format:	string
Units:	--
Minimum value:	0
Maximum value:	10
Default value:	007 (not defined)
Access:	read / write in any Sercos phase 2

C-0-0406 Hardware ID of EtherNet 1 Interface (GPP only)
C-0-0417 Hardware ID of EtherNet 2 Interface

This parameter displays the current optional Ethernet interface hardware typecode. When no Ethernet interface is available, the parameter displays *<no eth present>*.

C-0-0406 & C-0-0417 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	current Ethernet interface typecode if installed
Access:	read-only

C-0-0407 EtherNet 1 Interface Firmware Version (GPP only)
C-0-0418 EtherNet 2 Interface Firmware Version

This parameter displays the current optional Ethernet interface firmware version. When no Ethernet interface is installed, the parameter displays *<no eth present>*.

C-0-0407 & C-0-0418 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	current Ethernet firmware version if installed
Access:	read-only

C-0-0408 Ethernet 1 Driver Version (GPP only)
C-0-0419 Ethernet 2 Driver Version

This parameter displays the current optional Ethernet interface driver version. If no Ethernet interface is installed, the parameter displays *<no eth present>*.

C-0-0408 & C-0-0419 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	current Ethernet CIF driver if installed
Access:	read-only

C-0-0409 MAC Address of EtherNet 1 Interface (GPP only)
C-0-0420 MAC Address of EtherNet 2 Interface

The parameter displays the MAC address of the Ethernet Interface. The following is an example of the format “00-60-34-00-93-57”. The CIF05 library function cif_GetEthMacAddress() is used to read string.

C-0-0409 & C-0-0420 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0
Access:	read / write in any Sercos phase

C-0-0410 Fieldbus Slave Cyclic Channel Status (GPP only)

The parameter displays the status of the fieldbus cyclic channel. The following are possible displays:

- "NOT VALID:" – Not fieldbus card is availible or installed.
- "FATAL ERR: " – Fatal error while initalizing the fieldbus interface.
- "RUN: cyclic data exchange on bus" – Cyclic data exchange is valid.
- "INIT:" – Fieldbus interface has been intialized.
- "NOT INIT:" – Fieldbus interface has not been intialized.

C-0-0410 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0
Access:	read / write in any Sercos phase

Initialization Task Parameters (C-0-0503 through C-0-0537)

C-0-0503 Init. Task Instruction Rate

This parameter represents the number of instructions executed for the Initialization Task per Sercos Update. For example if the user sets C-0-0503 to 11, then the Initialization Task will execute 11 instructions before allowing lower priority tasks such as Serial Communications to execute. This parameter allows the user to balance the system between Initialization Task performance and overall system responsiveness.

C-0-0503 Attributes

Data length:	
Data type:	
Display format:	
Units:	--
Minimum value:	1
Maximum value:	24
Default value:	16
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-0522 Init. Task Diagnostic Message

This parameter displays the current diagnostic message and/or code for the Initialization task. During normal operation, a **Msg1** icon in the user program sets this message. If an error occurs during task execution, this diagnostic message is overwritten with an error message. For a complete listing of all system status and error codes, refer to the *Rexroth VisualMotion 11 Troubleshooting Guide*.

C-0-0522 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	read by control
Access:	read-only

C-0-0523 Init. Task Status Message

This parameter displays the current status message for the initialization task. A **Msg1** icon in the user program sets this message as an aid to the operator or for debugging purposes. This message is not overwritten with an error message, allowing debugging of an error condition set in the Init. Task Diagnostic Message.

C-0-0523 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	read by control
Access:	read-only

C-0-0530 Init. Task Current Instruction Pointer

This parameter returns a hexadecimal value equal to the initialization task's execution address (i.e. the instruction pointer). The hex value is an offset from the start of the program. This parameter is primarily used for debugging and troubleshooting programs.

For example:

"0x000000F0" indicates that the program counter is at 0xF0, or 240 bytes from the start of the program.

C-0-0530 Attributes

Data length:	4 byte data
Data type:	unsigned integer
Display format:	hexadecimal
Units:	--
Minimum value:	0x00000000
Maximum value:	0xFFFFFFFF
Default value:	read by control
Access:	read-only

C-0-0531 Init. Task Current Instruction

This parameter displays the mnemonic for the current instruction and the first 2 arguments of the instruction. The mnemonic is in the base code format generated by the control's compiler. This parameter is primarily used for debugging and troubleshooting programs.

C-0-0531 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	read by control
Access:	read-only

C-0-0532 Init. Task Pointer at Error

This status parameter displays the instruction pointer where the last task error occurred.

C-0-0532 Attributes

Data length:	4 byte data
Data type:	unsigned integer
Display format:	hexadecimal
Units:	--
Minimum value:	0x00000000
Maximum value:	-0xFFFFFFFF
Default value:	read by control
Access:	read-only

C-0-0533 Init. Task Composite Instruction Pointer

This parameter dynamically displays a flag and an instruction pointer indicating the relative memory address where a program instruction is being executed. This parameter is used by VisualMotion Toolkit to determine which icon to highlight in the Initialization task when using *Show Program Flow*.

Note: The highlighting of icons in nested subroutines with in the Init. task, when using *Show Program Flow*, is limited to 10 levels.

C-0-0533 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	read by control
Access:	read-only

C-0-0535 Init. Task Current Subroutine

This parameter indicates the current subroutine being executed with the function number and name. If function number and name information is not included in the user program file, the string “NONE” is returned.

C-0-0535 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	read by control
Access:	read-only

C-0-0536 Init. Task Stack Variable Data

This parameter displays the current stack local variable data. Stack variables are valid only while the program flow is within a task or subroutine. Maximum number of stack variables is 16. If there are no arguments or local variables in a task or function, the string “NONE” is returned.

C-0-0536 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	NONE
Access:	read-only

C-0-0537 Init. Task Subroutine Breakpoint

This parameter is reserved for future development.

BTC06 Teach Pendant (C-0-0801 through C-0-0814)

C-0-0801 Pendant Protection Level 1 Password

This parameter defines a four-digit numeric password that prevents entry into protected menus. If set to 0, the password is disabled.

Note: If the password is used, the pendant protection level bits (Register 1, bits 15 and 16) do not function.

C-0-0801 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	9999
Default value:	0
Access:	read / write in any Sercos phase

C-0-0802 Pendant Protection Level 2 Password

This parameter is reserved for future development.

C-0-0803 Pendant User Accessible Floats Section

This parameter defines the maximum allowable range for program floats to be user accessible from the BTC06. The operator can view all the program floats, but the operator can only access the program floats, up to the number set in this parameter. If the operator needs to change a program float greater than the number in this parameter, then the operator can either enter a password or set the pendant level protection bits (System Control Register 1, bits 15 & 16).

For example:

User Accessible Program Float Section = 10

When the operator selects Table Edit Menu/Float Table Menu, the operator can only access the first ten floats. The programmer is responsible for structuring the program floats properly. The allowable selections are as follows:

- 0 = no program floats are accessible
- -1 = all program floats are accessible
- n = number of defined program floats

Rexroth Bosch Group

C-0-0803 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	-1
Maximum value:	32767
Default value:	-1
Access:	read / write in any Sercos phase

C-0-0804 Pendant User Accessible Integers Section

This parameter defines the maximum allowable range for program integers to be user accessible from the BTC06. The operator can view all the program integers, but the operator can only access the program integers, up to number set in this parameter. If the operator needs to change a program integer greater than the number in this parameter then the operator can either enter a password or set the pendant level protection bits (System Control Register 1, bits 15 & 16).

For example:

User Accessible Program Integer Section = 10

When the operator selects Table Edit Menu/Integer Table Menu, the operator can only access the first ten integers. The programmer is responsible for structuring the program integers properly. The allowable selections are as follows:

- 0 = no program integers are accessible
- -1 = all program integers are accessible
- n = number of defined program integers

C-0-0804 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	-1
Maximum value:	32767
Default value:	-1
Access:	read / write in any Sercos phase

C-0-0805 Pendant Start of User Accessible Registers

This parameter defines the starting register that is accessible to the operator. The operator can view all the registers, but the operator can only access the registers beginning with C-0-0805 and ending with C-0-0806. If the operator needs to change a bit in a register outside the window of this parameter, then the operator can either enter a password or set the pendant level protection bits (System Control Register 1, bits 15 & 16). When the Register I/O Menu is selected on the BTC06, the first register to be displayed is the number stored in the Start of User Accessible Registers parameter.

C-0-0805 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	1024
Default value:	1
Access:	read / write in any Sercos phase

C-0-0806 Pendant End of User Accessible Registers

Refer to the description of *C-0-0806 Pendant End of User Accessible Registers* for details.

C-0-0806 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	1024
Default value:	1024
Access:	read / write in any Sercos phase

C-0-0807 Pendant Password Timeout

This parameter sets a timeout on the BTC06 password when using VT100 terminal firmware. After the password is entered (C-0-0801 Pendant Protection Level 1 Password), the user can access any screens requiring the password for the time set in this parameter. When a key is pressed, the timer is reset. After the timer expires, the password is again required. If the timeout is set to 0, the password is always required.

C-0-0807 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	seconds
Minimum value:	0 (disabled)
Maximum value:	3600 (one hour)
Default value:	30
Access:	read / write in any Sercos phase

C-0-0810 TPT Message and Prompt Control Word

This parameter is used to display a user task status message in the top two lines of the BTC06 and it can prompt the user for data entry into a variable. This parameter is used for the BTC06 when using VT100 terminal firmware.

```
0000000000000000
               |   Bit 1

Bits   1-8: Variable ID number
Bits  9-11: Variable type
Bits 12-14: Task ID
Bits 15-16: Control and Status
```

Fig. 15-5: Bit Description C-0-0810

C-0-0810 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0000000000000000
Access:	read / write in any Sercos phase

Bits 1-8: Variable ID Number

This ID number identifies the variable as a binary bit number between 0 and 255.

For example:

175 (decimal) is set as 10101111 (binary)

Bits 9-11: Variable Type

These bits identify the variable type for the number set in bits: 1-8. The following variable types are supported:

Bits 9-11	Value	Description
000	0	Integer Variable
001	1	Float Variable
010	2	Global Integer Variable
011	3	Global Float Variable

Table 15-71: TPT Variable Type Bits

Bits 12-14: Task ID

These bits identify which user task is associated with the selected variables. The following selections are available:

Bit 12-14	Value	Description
000	0	Inactive (no messaging)
001	1	Use task A status message
010	2	Use task B status message
011	3	Use task C status message
100	4	Use task D status message

Table 15-72: TPT Task ID Bits

Bits 15-16: Control and Status

These bits set the control and status displayed on the BTC06.

Bit 15-16	Value	Description
00	0	Done (status)
01	1	New prompt request for input (prompt for data entry) or data entry abort (when bits 1-8 = 0 and bits 9-11 = 0)
10	2	Operator input active (status)
11	3	Operator input error (status)

Table 15-73: TPT Control and Status Bits

VisualMotion Programming

Use the **Msg1** icon to create the message to be displayed on the first two lines of the BTC06.

Use the **Calc3** icon to build an integer that corresponds to the desired bit settings for parameter C-0-0810.

Use the **Param1** icon to transfer the integer to parameter C-0-0810. The integer will be converted to a 16-bit binary word and processed in this parameter.

Note: The programmer should set up the entire word before writing it to the parameter.

The task's status/prompt message appears on the first two lines of the BTC06 display. The programmer is responsible for formatting the message and accounting wrap-around. The BTC06 task keeps track of message changes and updates the display accordingly.

A task's status/prompt message may contain only one data entry field. Field size and location are determined by the following conventions:

Data entry fields are indicated by one or more '#' signs. Field size is determined by the number of consecutive '#' signs used.

Data entry fields may contain a decimal sub-field.

For example:

#####.##

Searching left to right, the first '#' sign found determines the field's location.

For example:

"Please enter part #: #####". The first '#' sign ("part #") would erroneously be used for data entry.

The absence of '#' signs in a prompt message forces a default field of 12 characters at the end of a message.

In the displayed message, '#' signs are blanked out and the cursor is placed left justified in the field.

When writing the message in the task, the first line can be terminated by using the '~' character. This makes for easy formatting of a message with a prompt on the second line.

Note: The functionality of the Message Setup Box remains unchanged. A variable can still be displayed in the message by using '%s' in a message. A variable that requires data entry must be entered as a series of '#' signs.

Note: The variable defined by the values in bits 1-8 and bits 9-11 is displayed in the task status message. If the variable number and type are not defined, a formatting string (consisting of one or more '#' signs) that is entered in the **Msg1** Icon will be displayed as '#' sign(s) instead of as the variable.

Control and Status

Prompt for user input

When the BTC06 task sees the new prompt status (binary 01 in bits 15 and 16), it prompts the operator for data entry. As soon as the operator begins to enter a value, the BTC06 sets the status to a binary 10 in bits 15 and 16 indicating that the operator is in the process of entering numerical data. No additional messages can be placed until the data entry is complete (press the OK or ESC key) or a Data entry abort is commanded. When the operator presses the OK or ESC key, the BTC06 task sets the status to a binary 00, (Done in bits 15 and 16), indicating that data input is complete and successful (no input error). The programmer should then check the input against internal maximum and minimum values.

Data entry abort

If desired, the programmer can use the **Param1** icon to force an Abort Message by setting a binary 01 status in bits 15 and 16 with the variable number and type set to 0. This abort can override the user input when a certain condition is met in the system (e.g. a hazardous condition).

Data input error

When a data input error occurs, the BTC06 flashes the data field for 3 seconds. Bits 15 and 16 are set to a binary 11, indicating the data input error, and the variable remains unchanged.

C-0-0811 User Task Controlled Menu ID for TPT

This parameter allows the programmer to control or “force” which menus the user can access during a task. This parameter is used for the BTC06 when using VT100 terminal firmware. The allowable selections are as follows:

0 = Inactive	10 = Global Float Table Edit Menu
1 = Main menu	11 = Jog Menu
2 = Program Menu	12 = Control Menu
3 = Table Edit Menu	13 = Register I/O Menu
4 = Absolute Table Edit Menu	14 = Parameter Menu
5 = Relative Table Edit Menu	15 = Card Parameter Menu
6 = Event Table Edit Menu	16 = Task Parameter Menu
7 = Integer Table Edit Menu	17 = Axis Parameter Menu
8 = Float Table Edit Menu	18 = Drive Parameter Menu
9 = Global Integer Table Edit Menu	19 = Diagnostic Menu

Table 15-74: TPT User Menu ID's

C-0-0811 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	19
Default value:	0
Access:	read / write in any Sercos phase

The BTC06 task continually scans parameter C-0-0811 for a non-zero value. As long as this parameter remains, zero, user tasks have no menu controls over the BTC06. However, when this parameter is set to a non-zero value, user tasks have complete control over BTC06 menus.

Writing a zero into C-0-0811 relinquishes user task control over BTC06 menus. The BTC06 continues to display the last active menu, but now the operator can freely select menus.

The programmer selects the desired active menu in a user task by writing a menu ID number into C-0-0811. At this point, the user can move only to adjoining menus lower in hierarchy, but not back to the higher menus.

For example:

If the control menu is “forced”, the user can move from there to the diagnostic menu, but not back to the main menu. To prevent movement between menus, the programmer can use Register 92-94 to mask the functionality of the F-keys.

VisualMotion Programming

For each menu to be “forced” during a task, the programmer writes the corresponding menu ID number into C-0-0811, using the **Param1** icon. The BTC06 task knows that a new menu request has been made by seeing the transition of C-0-0811 from its current value to a different value. The transition triggers the menu to change.

Note: A menu that may cause motion, such as the jog menu, checks that the selected task is stopped before motion is allowed. The error: “User task must be stopped” is issued if this is not the case (e.g., jogging is allowed only during manual or auto mode, but not while the task is running).

Motion menus check whether the task in the control has any motion queued to the path planner. For example, if a cycle stop is executed while motion is active, any attempt to jog results in the following error: “User task has motion pending." To allow jogging, all motion must be cleared from the path planner by switching to manual mode or performing a coordinated abort in the **Stop** Icon.

IMPORTANT: Parameter C-0-0811 must be active (≠0) for parameters C-0-0812, C-0-0813, and C-0-0814 to be functional.

IMPORTANT: System errors are handled in the same way, regardless of the user’s ability to control menu selection. System errors automatically transfer control to the diagnostic screen.

C-0-0812 User Task Controlled Task ID for TPT

This parameter allows the programmer to control or “force” which task motion system is displayed on the BTC06 for all axes and instructions defined in the active task. The user can also select the displayed task motion system by choosing it from the task menu. Using parameter C-0-0812, the programmer chooses the task for the user. This parameter is used for the BTC06 when using VT100 terminal firmware.

C-0-0812 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	0
Maximum value:	3
Default value:	0
Access:	read / write in any Sercos phase

Note: Parameter C-0-0811 must be active (≠0) for parameter C-0-0812 to be functional.

C-0-0813 User Task Controlled Axis Number for TPT

This parameter allows the programmer to control or “force” which single axis (defined by that axis’ Sercos address) is to be jogged. (The user can also select the axis by choosing it from the Jog Menu.) Using parameter C-0-0813, the programmer chooses the axis for the user. This parameter is used for the BTC06 when using VT100 terminal firmware.

If an invalid axis or point number is found, the BTC06 responds by issuing an error message. It is the programmer’s responsibility to ensure these parameters are set appropriately before taking control of a menu.

C-0-0813 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	1
Maximum value:	100
Default value:	1
Access:	read / write in any Sercos phase

Note: Parameter C-0-0811 must be active (≠0) for parameter C-0-0813 to be functional.

C-0-0814 TPT Data Transaction Word

This parameter allows the programmer to:

- monitor the user’s index value in a particular BTC06 menu (regardless of the user’s permission to change the field value).
- direct the user’s data entry in a menu (e.g., to direct the point to be taught in the Jog Menu, the register number in the Register Menu, the current point number in the Float Table Menu, etc.) with the ability to assign the user read-only (“lock”) or write (“set”) privileges.

```
0000000000000000
               |  Bit 1

Bits   9-1: Data index
Bits 14-10: Menu ID number
Bits 16-15: Control and Status
```

Fig. 15-6: Bit Description C-0-0814

C-0-0814 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0000000000000000
Access:	read / write in any Sercos phase

Bits 9-1: Data Index

The data index is dependent on the selected menu ID number. These bits can display the value for a selected menu.

For example:

If 13, Register menu is selected for bits 10-14, then bit 1-9 will display the register number being requested or being written in as a 16-bit binary word.

Bits 14-10: Menu ID Number

The following table lists the available menu selections along with its index type.

Bits 14-10	Menu No.	Menu Name	Index Type
00001	1	Main Menu	not defined
00010	2	Program Menu	line number
00011	3	Table Edit Menu	not defined
00100	4	Absolute Table Menu	point number
00101	5	Relative Table Menu	point number
00110	6	Event Table Menu	event number
00111	7	Integer Table Menu	integer number
01000	8	Float Table Menu	float number
01001	9	Global Integer Menu	global integer number
01010	10	Global Float Menu	global float number
01011	11	Jog Menu ABS Menu	point number
01100	12	Control Menu	not defined
01101	13	Register Menu	register number
01110	14	Parameter Table Select Menu	not defined
01111	15	Card Parameter	not defined
10000	16	Task Parameter	not defined
10001	17	Axis Parameter	not defined
10010	18	Drive Parameter	not defined
10011	19	Diagnostic Menu	not defined

Table 15-75: TPT User Menu ID Bits

Bits 16-15: Status and Control

These bits set the status and control for the BTC06 display.

Bit 16-15	Description
00	Done (status)
01	Read Request
10	Set/Unlock Command (write a value in a specific screen, editable by the user)
11	Lock Command (write a value in a specific screen, not editable by the user)

Table 15-76: TPT Control and Status Bits

VisualMotion Programming

Use the **Calc3** icon to build an integer that corresponds to the desired bit settings for parameter C-0-0814. Use the **Param1** icon to transfer the integer to parameter C-0-0814. The integer will be converted to a 16-bit binary word and processed in this parameter.

Note: The programmer should set up the entire word before writing it to the parameter.

The user task builds this word and sets the transaction request in bits 15 and 16. The transaction is complete when the Done status (bits 15-16 = 00) is set by the BTC06.

Note: If the absolute point number is “locked” in the Jog Menu after teaching a point, the BTC06 does not automatically advance the point number. The point number is always dictated by parameter C-0-0814.

Note: Parameter C-0-0811 must be active (≠0) for parameter C-0-0814 to be functional.

Generic Cases

The following table is a list of generic cases for this parameter.

Bits 16-15	Bits 14-10	Bits 9-1	Action
10	0	0	clears all locks in all menus
10	≠0	0	clears locks in a specific menu
11	0	0	locks all data indexes in all menus
11	≠0	≠0	locks data index in a specific menu
11	0	≠0	locks data index in the current menu
10	≠0	≠0	sets data index in a specific menu
10	0	≠0	sets data index in the current menu
01	0	0	requests data index from current menu the user is viewing
01	≠0	0	requests data index from a specific menu

Table 15-77: Generic Cases for TPT Control and Status Bits

System Memory Parameters (C-0-0993 and C-0-0996)

C-0-0993 Software Reset for PPC

This parameter is used to reset power to the control. Entering "YES" in this parameter will cause the control to reset (cycle power) and reinitialize after a 2-second delay. The control must first be switched to parameter mode before this parameter can be accessed.

Note: Although the PPC-R22.1 has a hardware reset button (S2), this parameter resets the control while in parameter mode. Resetting the PPC-R22.1 using the S2 button will cause an immediate reset regardless of motion. The PPC-P11.1 does not have a hardware button and can only be reset using this parameter.

C-0-0993 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	---
Minimum value:	---
Maximum value:	---
Default value:	NO
Access:	read in any Phase/ write in Phase 2

C-0-0994 Shutdown Command for Flash Programming

This parameter is used to switch the control to download mode (DL). This mode is necessary when upgrading control or drive firmware using Dolfi software via a RS232 serial interface. Entering "YES" while in parameter mode, switches the control to download mode and displays the characters DL in the control’s H1 display. Cycle power to the control to restore it to normal operating mode.

Note: The control can also be switched to DL mode by cycling power to the control while holding the S1 button until the bootloader firmware version is displayed.

C-0-0994 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	NO
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-0996 Clear Program and Data Memory

When a "YES" is written to this parameter, the control's non-volatile memory will be cleared, including all parameters, programs, and PLC data.

Note: Use the archive function, under menu selection ***Commission*** ⇒ ***Archive***, before clearing the control's memory. Once program and data memory is cleared, it can no longer be retrieved.

C-0-0996 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	NO
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-0997 Clear Diagnostic Log

This parameter is used to clear the diagnostic log. When "YES" is written to C-0-0997, the Diagnostic log is cleared.

C-0-0997 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	NO
Access:	read in any Sercos phase / write in Sercos phase 2

Integrated PLC Parameters (C-0-1600 through C-0-1639)

C-0-1600 Integrated PLC: Timeslice of Sercos Cycle Time

This parameter sets the allocated time for the integrated PLC of the Sercos cycle time. For example, if the user sets C-0-1600 to 25%, and the Sercos cycle time is 2ms, then the Tasks for PLC will be active for 0.5ms.

C-0-1600 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	percent
Minimum value:	10
Maximum value:	75
Default value:	25
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-1601 Integrated PLC: Configuration

This parameter is the configuration word for the PLC interface. The following bits are supported:

Bits 2-1: PLC Boot Behavior

The following bit settings configure the PLC boot up behavior:

Bit 2	Bit 1	Description
0	0	Standard boot behavior. The PLC starts if the system was running when the control was shutdown. Otherwise the PLC remains stopped.
1	1	Standard boot behavior as described above when bits 1 and 2 are set to 0.
0	1	The PLC starts if the boot project is available.
1	0	The PLC remains stopped.

Table 15-78: PLC Boot Behavior

Bit 8: PLC Error Reaction

The following bit setting configures the PLC error reaction:

Bit 8	Description
0	A fatal error is generated in the motion system if a PLC error shuts down the PLC tasks.
1	PLC errors are ignored by the motion system.

Table 15-79: PLC Error Reaction

Bit 9: I/O Image Output State

The following bit setting configures the I/O image output state behavior:

Bit 9	Description
0	I/O image outputs remain in their previous state when the PLC is stopped.
1	All I/O image outputs are set to 0 when the PLC is stopped.

Table 15-80: I/O Image Output State

Bit 11: Ignore PLC I/O Image Update Errors

The following bit setting configures the PLC I/O Update error indication:

Bit 11	Description
0	PLC I/O Update errors are issued
1	PLC I/O Update errors are ignored

Table 15-81: Ignore PLC I/O Image Update Errors

C-0-1601 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	11111111 11111111
Default value:	00000000 00000000
Access:	read in any Sercos phase / write in Sercos phase 2
Activation:	requires power cycle

C-0-1602 Integrated PLC: Control Word Status

This parameter is used to monitor the error status of the Integrated PLC control word C-0-1603.

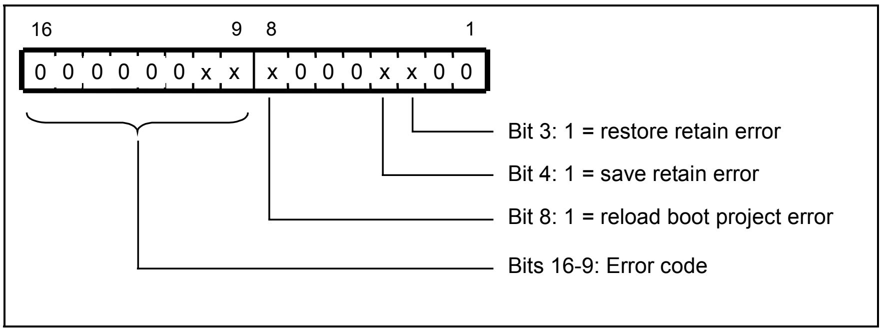


Fig. 15-7: Integrated PLC: Control Word Bit Description

Rexroth
Bosch Group

Bit	Status	Description
3	Restore Retain Error	An error occurred while restoring the retain data
4	Save Retain Error	An error occurred saving the retain data
8	Reload Boot Project Error	The boot project did not load
16 - 9	Error Code	These bits display the error codes for the PLC control word

Table 15-82: Integrated PLC: Control Word Status

Error codes are only available for bits 3 and 4. The error code displayed in bits 9 and 10 is valid for the currently set bit in either bit 3 or bit 4.

C-0-1602 Attributes

Data length:	2 byte data
Data type:	binary word
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 11111111
Default value:	00000000 00000000
Access:	read / write in any Sercos phase

C-0-1603 Integrated PLC: Control Word

This control word is used to initiate the following Integrated PLC commands:

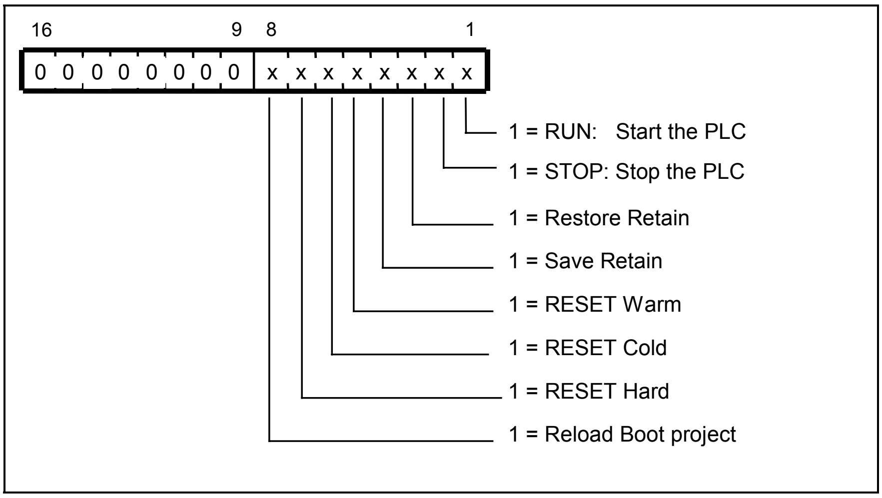


Fig. 15-8: Integrated PLC: Control Word Bit Description

Bit	Command	Description
1	Run	The PLC is started, if a program is loaded. The status word displays RUN.
2	Stop	The PLC is stopped. The tasks stop immediately in their cycle, and do no restart. The status word displays STOP.
3	Restore Retain	The retain data of the PLC is restored from the file ‘retain.bin’, if it exists on the control.
4	Save Retain	The retain data of the PLC is saved to the file ‘retain.bin’ on the control.
5	RESET Warm	The PLC is reset without initializing the retain variables. The status word displays STOP.

Bit	Command	Description
6	RESET Cold	The PLC is reset and the retain variables are initialized. The status word displays STOP.
7	RESET Hard	All of the data of the PLC including the boot project is deleted. The status word displays STOP.
8	Reload Boot project	The boot project is loaded. After a reboot, the PLC behaves based on the settings of bits 1 and 2 in parameter C-0-1601.

Table 15-83: Integrated PLC: Control Word

Each command is initiated with the rising edge of the bit. If multiple bits are selected, they are processed from most significant to least significant bits.

C-0-1603 Attributes

Data length:	2 byte data
Data type:	binary word
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 11111111
Default value:	00000000 00000000
Access:	read / write in any Sercos phase

C-0-1604 Integrated PLC: Status Word

This parameter is displays the current status of the PLC. The following bits display the current PLC status.

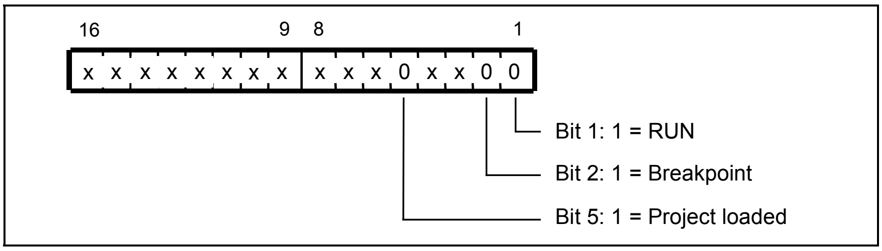


Fig. 15-9: Integrated PLC: Status Word

Bit	Status	Description
1	RUN	The PLC is running.
2	Breakpoint	The PLC is stopped at a breakpoint.
5	Project loaded	A project is loaded in RAM

Table 15-84: Integrated PLC: Status Word

C-0-1604 Attributes

Data length:	2 byte data
Data type:	binary word
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 11111111
Default value:	00000000 00000000
Access:	read / write in any Sercos phase

C-0-1605 Integrated PLC: Project Name

The parameter displays the name of the current PLC project. (i.e., SamplePLCProgram.pro). After a hard reset, the old name is still displayed.

C-0-1605 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0
Access:	read-only

C-0-1610 Integrated PLC: ELS System Association

This parameter identifies if the ELS system is associated to task A or to the PLC. If set to 0, the ELS system is associated to Task A. If set to 1, the ELS system is associated to the PLC. This option is set under ***Setup ⇒ Processes ⇒ ELS***.

C-0-1610 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	0
Maximum value:	1
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-1611 Integrated PLC: Diagnostic Code

This parameter displays the active system status or error code issued by the Integrated PLC. The following are Integrated PLC diagnostic codes:

Code	Description
6001	PLC Running
6002	PLC Stopped or PLC Stopped at Breakpoint
6011	ERROR_INI_VM_PROG_STOP_PHASE4 The PLC program was stopped by the user while the control was in Sercos phase 4.
6012	ERRORS_INI_VM_GENERAL This is a general error. Refer to C-0-1613
2000	ERRORS_INI_P3S_FATAL_COM An error occurred while handling Integrated PLC communication.
2001	ERRORS_INI_P3S_FATAL_SEM An internal fatal error occurred while initializing a function block.
2002	ERRORS_INI_P3S_FATAL_MEM An internal fatal error occurred while initializing a function block.
2003	ERRORS_INI_P3S_MSG_PLCCF IndraLogic contains a PLC configuration error.
2004	ERRORS_INI_P3S_FILE_ERR An error occurred while accessing the file system.
2005	ERRORS_INI_P3S_FATAL_TSK An internal fatal error occurred in the initialization of the Integrated PLC.
16	RTSEXCPT_WATCHDOG The IndraLogic software watchdog PLC task expired.
19	RTSEXCPT_PROGRAMCHECKSUM A checksum error occurred after a PLC program was downloaded.
20	RTSEXCPT_FIELDBUS_ERROR An error occurred during the fieldbus master initialization.
21	RTSEXCPT_IOUPDATE_ERROR An error occurred while accessing data for the IO image.

Table 15-85: Integrated PLC Diagnostic Codes

C-0-1611 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	6999
Default value:	Highest priority code is displayed
Access:	read-only

C-0-1612 Integrated PLC: Diagnostic Message

This parameter displays the active system status or error message issued by the Integrated PLC for the corresponding error code in C-0-1611.

C-0-1612 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	0
Maximum value:	80
Default value:	Highest priority message is displayed
Access:	read-only

C-0-1613 Integrated PLC: Extended Diagnostic

This is a dynamic Integrated PLC message used to provide additional diagnostic information for the status warning or error message C-0-1612.

C-0-1613 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	0
Maximum value:	80
Default value:	dependent on C-0-1612
Access:	read-only

C-0-1620 Integrated PLC: POU Error Log List

VisualMotion maintains a log of the last 30 errors encountered in the BRC supplied function blocks. The data included for each entry is based on the error structure provided in the CommonTypes.lib file. The IndraLogic user is provided a function block to convert errors encountered in user defined function block into the ERROR_STRUCT format and then to insert them into this diagnostic log.

Each diagnostic log string includes:

- the date and time that the error occurred (ERROR_STRUCT.TimeStamp)
- POU type (ERROR_STRUCT.PouType) –
- Pointer to the POU instance returning the error (ERROR_STRUCT.Instance)
- The object associated with the error (ERROR_STRUCT.Adr2)
- Error code (ERROR_STRUCT.Code)
- Error description (ERROR_STRUCT.Additional)

Error structure elements that are not included:

- Source – NOT USED
- Class – Fixed at Error_1_Error_Class for all errors
- ADR1 – Fixed at Local_Control for all errors

Format:

```
11-03 12:15:47 80040001 00FC90E8 1 3 05000006
  |      |        |        |     | |       |__ Description
  |      |        |        |     | |      (0x5000006 = Position feedback #2
  |      |        |        |     | |       is not configured in the AT)
  |      |        |        |     | |
  |      |        |        |     | |____ Error Code (3 = ACCESS_ERROR_CODE)
  |      |        |        |     |
  |      |        |        |     |______ Object causing error (1 = Axis_1)
  |      |        |        |
  |      |        |        |_______ FB instance Address
  |      |        |                (pointer to fb instance)
  |      |        |
  |      |        |_______________ POU Type (80040001 = MV_ReadActualVelocity)
  |      |
  |      |________________________ time error occurred (From TimeStamp)
  |
  |________________________________ date error occurred (From TimeStamp)
```

C-0-1620 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	0
Maximum value:	30
Default value:	current FB error log list
Access:	read-only

C-0-1621 Integrated PLC: POU Error Log List Options

When bit 1 is set, the raising edge clears the POU error log list in C-0-1620. Following the command, the bit is then automatically cleared.

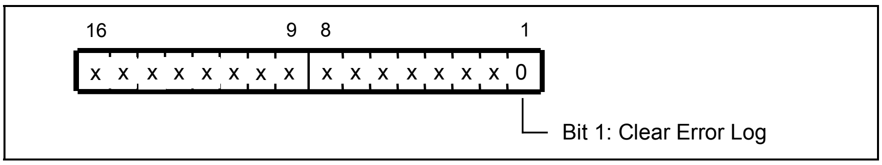


Fig. 15-10: Integrated PLC: POU Error Log List Options

C-0-1621 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	00000000 00000000
Access:	read / write in any Sercos phase

C-0-1638 Integrated PLC: SysLibDirect Read Access Parameters

This parameter contains a list of allowable VisualMotion control, axis and task parameters that can be read using the SysLibDirect output data access.

C-0-1621 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	list of supported parameters
Access:	read-only

C-0-1639 Integrated PLC: SysLibDirect Write Access Parameters

This parameter contains a list of allowable VisualMotion control, axis and task parameters that can be written to by the SysLibDirect output data access.

C-0-1639 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	list of supported parameters
Access:	read-only

System Parameter Lists (C-0-2000 through C-0-2021)

C-0-2000 List of All Parameters

This parameter contains a list of all control parameters that are part of the current firmware version. The contents of this parameter can be viewed by double clicking on the parameter number in the *Parameter Overview* tool.

C-0-2000 Attributes

Data length:	variable length 4 byte data
Data type:	unsigned integer
Display format:	IDN (parameter number)
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

C-0-2001 List of Required Parameters

This parameter contains a list of all required control parameters that are part of the archive / restore function. The contents of this parameter can be viewed by double clicking on the parameter number in the *Parameter Overview* tool.

C-0-2001 Attributes

Data length:	variable length 4 byte data
Data type:	unsigned integer
Display format:	IDN (parameter number)
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

C-0-2002 List of Invalid A-, C- and T- Parameters

This parameter displays a list of all parameters with an invalid CRC (**C**yclic **R**edundancy **C**heck). The maximum number of entries is fixed to 200. If more than 200 parameters are invalid, they will not be displayed. When switching to Sercos phase 4, an error is issued until all invalid parameters in the list have been corrected. When the user clears the error, another message is displayed: "009 Select Parameter Mode to Continue", indicating that the user has to switch back to P2. The list contains entries to identify the wrong parameters in the following format:

pxx: p-0-yyyy where p is the parameter type (A-, C, or T-parameters),

xx is the drive address (1 to 64), task number (1 to 4) or in case of a C-parameter it is always "00"

yyyy is the parameter number.

For example:

"A03: A-0-0005" specifies drive number 3, axis parameter A-0-0005.

C-0-2002 Attributes

Data length:	max 200 string entries
Data type:	string array
Display format:	string
Units:	--
Minimum value:	0
Maximum value:	200
Default value:	0
Access:	read-only

C-0-2010 List of Sercos Devices

During Sercos phase 1 initialization, the control scans the Sercos ring for all connected Sercos devices. These devices include all drives that are connected via the Sercos ring and any Sercos I/O stations. The devices found will be listed in sequential order by Sercos address. The contents of this parameter can be viewed by double clicking on the parameter number in the *Parameter Overview* tool.

C-0-2010 Attributes

Data length:	variable length 2 byte data
Data type:	integer
Display format:	unsigned decimal
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

C-0-2011 List of Sercos Drives

During Sercos phase 1 initialization, the control scans the Sercos ring for all connected Sercos digital drives. These digital drives include any Rexroth drives that are connected via the Sercos ring. The digital drives found will be listed in sequential order by Sercos address. The contents of this parameter can be viewed by double clicking on the parameter number in the *Parameter Overview* tool.

C-0-2011 Attributes

Data length:	variable length 2 byte data
Data type:	integer
Display format:	unsigned decimal
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

C-0-2012 List of Sercos I/O Stations

During Sercos phase 1 initialization, the control scans the Sercos ring for all connected Sercos I/O devices. The devices found will be listed in sequential order by Sercos address. The contents of this parameter can be viewed by double clicking on the parameter number in the *Parameter Overview* tool.

C-0-2012 Attributes

Data length:	variable length 2 byte data
Data type:	integer
Display format:	unsigned short
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

C-0-2013 I/O Configuration List

During Sercos phase 1 initialization, the control scans the Sercos ring for any Sercos I/O modules. The devices found will be listed in sequential order by Sercos address. The contents of this parameter can be viewed by double clicking on the parameter number in the *Parameter Overview* tool.

C-0-2013 Attributes

Data length:	variable length 2 byte data
Data type:	integer
Display format:	unsigned short
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

The elements in the Sercos I/O Configuration List are displayed using the following format:

```
CD 1.2013.1 100 11 2066 I 1 R1040 2
|__________|  |   |   |   |  |  |    |_ Slot number
     |        |   |   |   |  |  |
              |   |   |   |  |  |_____ Module identifier
                                       (R=RECO,I=Inline)
 Protocol     |   |   |   |  |_ Number of registers
  Header      |   |   |   |      (1,2,3or4)
              |   |   |   |
              |   |   |   |___ IO type
              |   |   |       (I=input,O=output)
              |   |   |
              |   |   |_______ Sercos IDN
              |   |             [P(number-32768)]
              |   |
              |   |___________ Sercos drive address
              |
              |______________ Register number assigned
```

C-0-2017 I/O User Configuration List

This parameter is used to store I/O devices configured using I/O Setup tool. RECO I/O modules (Local and Sercos) stored in parameter C-0-2013 I/O Configuration List as well as any drive I/O configured using the I/O Setup tool are stored in this parameter.

For detailed information on the *I/O Setup tool*, refer to the *Functional Description, section 7.2.*

C-0-2017 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-2019 Beckhoff I/O Configuration List

This parameter displays the current Beckhoff I/O module configuration installed in the system along with the system assigned register. The PPC Control automatically configures the Beckhoff I/O station for strict data configuration.

The following format is used to identify each module:

```
CD 1.2019.1  11  8192  Digital Input Channel  700  2
     |        |    |             |             |   |_ Number of
     |        |    |             |             |      Registers Used
 Protocol     |    |             |             |
  Header      |    |             |             |______ PPC Assigned
              |    |             |                     Register
              |    |             |
              |    |             |____ Module Description
              |    |
              |    |_______ Sercos IDN
              |
              |___________ Sercos drive address
```

C-0-2019 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

C-0-2020 Diagnostic Log List

VisualMotion maintains a log of the last 100 diagnostic errors. Each diagnostic log string includes the date and time that the error occurred, the shutdown error code, and any other error codes. The most recent error is listed first.

Format:

```
11-03 12:15:47 420 1 28 0
                          |_ extended secondary error code

                       |___ secondary error code
                            (28 = excessive deviation)

                    |______ extended error code (1=drive)

                |__________ error code
                            (420=Drive D shutdown error)

         |_________________ time error occurred
                            (refer to C-0-0126 for time
                             and date)
  |________________________ date error occurred
```

The extended secondary code is the data that varies in a message, such as the drive number in “Drive D Shutdown Error." If there is a task or drive error, it is printed as a secondary error code. Error codes are not supported on all versions of drives.

Emergency Stop and warnings are normally not included in this list. To log these errors, set options in parameter C-0-2021 Diagnostic Log Options.

C-0-2020 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	current diagnostic log list
Access:	read-only

C-0-2021 Diagnostic Log Options

This parameter sets which errors should be log and displayed in parameter C-0-2020 Diagnostic Log List. The options that can be active in this parameter are list in the following 16-bit word.

```
0000000000000000
               |  Bit 1

Bit 1: Include E-Stop errors (400)
Bit 2: Include warnings
Bit 3: Include drive undervoltage errors (26)
```

Fig. 15-11: Bit Description C-0-2021

C-0-2021 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0000000000000100 (drive undervoltage error)
Access:	read / write in any Sercos phase

C-0-2022 Probe Exception Handler

This parameter is used for internal purposes and does not provide any functionality to the user.

Oscilloscope Parameters (C-0-2501 through C-0-2523)

The oscilloscope tool is launched by selecting ***Diagnostics*** ⇒ ***Oscilloscope***. The parameterization of the oscilloscope function is found in this section.

- Signal selection parameterization
- Signal timing
- Oscilloscope trigger and control
- Oscilloscope data

C-0-2501 Oscilloscope Signal 1 Type

Parameters C-0-2501, C-0-2502, C-0-2503, and C-0-2524 identify the data type (e.g., axis parameter) for the 4 oscilloscope signals. The following table lists the supported data types:

Value in C-0-2501 *	Data Type	Valid for GPP 11	GMP 11
0	None	X	X
1	Program float variable (Fx)	X	X
2	Program integer variable (Ix)	X	X
3	Global float variable (GFx)	X	X
4	Global integer variable (GIx)	X	X
5	Axis parameter	X	X
6	Register bit	X	X
7	+/- Register	X	X
8	Control parameter	X	X
9	Task parameter	X	X
* also valid for C-0-2502, C-0-2503, and C-0-2524			

Table 15-86: Oscilloscope Signal Type

C-0-2501 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	9
Default value:	3
Access:	read / write in any Sercos phase

C-0-2502 Oscilloscope Signal 2 Type

Refer to *C-0-2501 Oscilloscope Signal 1 Type* for a description of this parameter.

C-0-2503 Oscilloscope Signal 3 Type

Refer to *C-0-2501 Oscilloscope Signal 1 Type* for a description of this parameter.

C-0-2504 Oscilloscope Signal 1 ID Number

This parameter identifies the ID number for the data type written in parameter C-0-2501 Oscilloscope Signal 1 Type. The value for this parameter is written using only the parameter number's significant digits (e.g., C-0-0083 would be written as 83 and F4 would be written as 4). This description is also valid for C-0-2505, C-0-2506, and C-0-2525.

The following table contains a list of valid control parameter ID numbers when signal type 8 is written in C-0-2501.

Valid Control Parameters		Valid for	
Value	**Description**	**GPP 11**	**GMP 11**
200	Current Load Due to Motion	X	X
201	Peak load Due to Motion	X	X
202	Current Load Due to I/O	X	X
203	Peak Load Due to I/O	X	X
3204	PMG 1 Current Peak Group Deviation	X	X
3205	PMG 1 Maximum Deviation	X	X
3214	PMG 2 Current Peak Group Deviation	X	X
3215	PMG 2 Maximum Deviation	X	X
3224	PMG 3 Current Peak Group Deviation	X	X
3225	PMG 3 Maximum Deviation	X	X
3234	PMG 4 Current Peak Group Deviation	X	X
3235	PMG 4 Maximum Deviation	X	X
3244	PMG 5 Current Peak Group Deviation	X	X
3245	PMG 5 Maximum Deviation	X	X
3254	PMG 6 Current Peak Group Deviation	X	X
3255	PMG 6 Maximum Deviation	X	X
3264	PMG 7 Current Peak Group Deviation	X	X
3265	PMG 7 Maximum Deviation	X	X
3274	PMG 8 Current Peak Group Deviation	X	X
3275	PMG 8 Maximum Deviation	X	X

Table 15-87: List of Valid Control Parameters for C-0-2504

The following table contains a list of valid task parameter ID numbers when signal type 9 is written in C-0-2501.

Valid Task Parameters		Valid for	
Value	**Description**	**GPP 11**	**GMP 11**
111	Current X Position	X	X
112	Current Y Position	X	X
113	Current Z Position	X	X

Table 15-88: List of Valid Task Parameters for C-0-2504

The following table contains a list of valid axis parameter ID numbers when signal type 5 is written in C-0-2501.

Valid Axis Parameters		Valid for	
Value	**Description**	**GPP 11**	**GMP 11**
100	Target Position	X	X
101	Commanded Position	X	X
102	Feedback Position	X	X
111	Commanded Velocity	X	X
112	Feedback Velocity	X	X
120	Programmed Acceleration	X	X
141	Torque Mode Commanded Torque	X	X
142	Torque Feedback (Cyclic)	X	X
156	Phase Offset Velocity Feedback	X	X
158	Relative Phase Offset Distance Remaining	X	X
190	Optional MDT Command 1	X	X
191	Optional MDT Command 2	X	X
192	Optional MDT Command 3	X	X
195	Optional AT Feedback 1	X	X
196	Optional AT Feedback 2	X	X

Table 15-89: List of Valid Axis Parameters for C-0-2504

C-0-2504 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	65535
Default value:	1
Access:	read / write in any Sercos phase

C-0-2505 Oscilloscope Signal 2 ID Number

This parameter identifies the ID number for the data type written in parameter C-0-2502 Oscilloscope Signal 2 Type. Refer to the tables in parameter C-0-2504 Oscilloscope Signal 1 ID Number for valid control, task, and axis parameters.

C-0-2506 Oscilloscope Signal 3 ID Number

This parameter identifies the ID number for the data type written in parameter C-0-2503 Oscilloscope Signal 3 Type. Refer to the tables in parameter C-0-2504 Oscilloscope Signal 1 ID Number for valid control, task, and axis parameters.

C-0-2507 Oscilloscope Signal 1 Axis Number

This parameter is only applicable when the signal type for C-0-2501, C-0-2502, C-0-2503, or C-0-2524 is set to 5, 6, or 9.

The following table lists the allowable values for this parameter based on supported signal type:

C-0-2501 Signal Type *	Allowable Value in 2507 * *	Description
5 (axis number)	1 - 64	value equals Sercos address for selected axis parameter in C-0-2504, C-0-2505, C-0-2506, or C-0-2525
6 (register bit)	1 -16	value equals bit number for selected register number in C-0-2504, C-0-2505, C-0-2506, or C-0-2525
9 (task)	1 = A 2 = B 3 = C 4 = D	value equals task letter for the selected task parameter in C-0-2504, C-0-2505, C-0-2506, or C-0-2525
* also valid for C-0-2502, C-0-2503, and C-0-2524 ** also valid for C-0-2508, C-0-2509, and C-0-2526		

Table 15-90: Oscilloscope Signal Axis Number

C-0-2507 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	1
Maximum value:	64
Default value:	1
Access:	read / write in any Sercos phase

Note: A value of 1 is written in this parameter when a signal type other than 5, 6, or 9 is used.

C-0-2508 Oscilloscope Signal 2 Axis Number

Refer to *C-0-2507 Oscilloscope Signal 1 Axis Number* for a description of this parameter.

C-0-2509 Oscilloscope Signal 3 Axis Number

Refer to *C-0-2507 Oscilloscope Signal 1 Axis Number* for a description of this parameter.

C-0-2510 Oscilloscope Sampling Rate

This parameter sets the sampling rate in milliseconds to determine how often a trace is captured. This parameter is used in conjunction with C-0-2514 Oscilloscope Sample Count to determine the capture duration for the entry trace. Sampling rate is set in the *Oscilloscope Options* window by selecting *Timing* from the oscilloscope main menu. Sample rates available when using the Oscilloscope are 2, 4, 8, 16, 32 and 64.

Note: The smallest sampling rate is limited by C-0-0099 Sercos Cycle Time.

For example:

Capture duration = [(C-0-2510 = 8ms) * (C-0-2514 = 500)] = 4000 ms

C-0-2510 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	ms
Minimum value:	1
Maximum value:	64
Default value:	2
Access:	read / write in any Sercos phase

C-0-2511 Oscilloscope Signal 1 List

This parameter stores all captured data for the configured signal 1. This information is uploaded from the control after the capture is complete.

C-0-2511 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

C-0-2512 Oscilloscope Signal 2 List

This parameter stores all captured data for the configured signal 2. This information is uploaded from the control after the capture is complete. Refer to *C-0-2511 Oscilloscope Signal 1 List* for the parameters attributes.

C-0-2513 Oscilloscope Signal 3 List

This parameter stores all captured data for the configured signal 3. This information is uploaded from the control after the capture is complete. Refer to *C-0-2511 Oscilloscope Signal 1 List* for the parameters attributes.

C-0-2514 Oscilloscope Sample Count

This parameter sets the quantity of data captured for each signal. This parameter is used in conjunction with C-0-2510 to determine the capture duration for the entry trace. Sample count is set in the *Oscilloscope Options* window by selecting *Timing* from the oscilloscope main menu. The available sample count when using the Oscilloscope are 100, 200, 500, 1000, 2000 and 4000.

C-0-2514 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	10
Maximum value:	4096
Default value:	10
Access:	read / write in any Sercos phase

C-0-2515 Oscilloscope Trigger Post-count

The oscilloscope utility can use an optional pretrigger value to display a percentage of the total sample count (C-0-2514 Oscilloscope Sample Count) before the actual configured signals are captured. A pretrigger can be set in the *Oscilloscope Options* window by selecting *Timing* from the oscilloscope main menu.

This parameter displays the remainder of the total sample count after the pretrigger percentage.

For example:

Trigger post count = (C-0-2514) * (1 - (pretrigger/100))

Pretrigger = 100 * (1 - (C-0-2515/C-0-2514))

C-0-2515 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	4096
Default value:	0
Access:	read / write in any Sercos phase

C-0-2516 Oscilloscope Trigger Type

When using the oscilloscope utility, a trigger can be user initiated or internally initiated. This parameter sets the signal type that will be used as the trigger enable. Refer to *C-0-2501 Oscilloscope Signal 1 Type* for available types. The signal type set in this parameter will trigger the oscilloscope after the polarity and threshold settings are true in the *Card Signal Setup* window. Select ***Signal Selection*** from the oscilloscope's main menu to display the *Card Signal Setup* window.

C-0-2516 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	9
Default value:	0
Access:	read / write in any Sercos phase

C-0-2517 Oscilloscope Trigger ID Number

This parameter identifies the numeric value that corresponds to the selected signal type in parameter C-0-2516 Oscilloscope Trigger Type. Refer to *C-0-2504 Oscilloscope Signal 1 ID Number* for details.

C-0-2517 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	65535
Default value:	0
Access:	read / write in any Sercos phase

C-0-2518 Oscilloscope Trigger Axis or Mask

This parameter is only applicable when C-0-2516 is set with one of the following trigger types:

C-0-2516 Signal Type	Allowable Value in 2518	Description
5 (axis number)	1 - 64	value equals Sercos address for selected axis parameter in C-0-2516
6 (register bit)	1 -16	value equals bit number for selected register number in C-0-2516
9 (task)	1 = A 2 = B 3 = C 4 = D	value equals task letter for the selected task parameter in C-0-2516

Table 15-91: Oscilloscope Signal Axis Number

C-0-2518 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	64
Default value:	0
Access:	read / write in any Sercos phase

Note: A value of 1 is defaulted in this parameter when a signal type other than 5, 6, or 9 is used.

C-0-2519 Oscilloscope Trigger Level or Mask

This parameter sets the level or threshold to which the oscilloscope will use its internal trigger and capture the configured signals in C-0-2516 Oscilloscope Trigger Type, C-0-2517 Oscilloscope Trigger ID Number and C-0-2518 Oscilloscope Trigger Axis or Mask. The trigger level can be a variable value, state of a register bit, float value, etc.

C-0-2519 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	--
Minimum value:	+ 0.0
Maximum value:	dependent on C-0-2516 Oscilloscope Trigger Type, C-0-2517, C-0-2518
Default value:	+0.0
Access:	read / write in any Sercos phase

C-0-2520 Oscilloscope Trigger Mode

This parameter sets the polarity or direction in which the trigger level set in parameter *C-0-2519 Oscilloscope Trigger Level or Mask* will be detected. The selections used in this parameter are:

- 1 = Positive
- 2 = Negative
- 3 = Positive or Negative

C-0-2520 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	1
Maximum value:	3
Default value:	1
Access:	read / write in any Sercos phase

C-0-2521 Oscilloscope Trigger Source

This parameter sets the trigger source that will be used by the oscilloscope utility for capturing signal traces. A trigger source can be initiated either by the user or internally. A user-initiated trigger is only active when the operator presses the enable trigger button  in the oscilloscope window. An internally initiated trigger is configured through oscilloscope parameters and active when all parameter requirements are met. The selections used in this parameter are:

- 1 = User Initiated
- 2 = Internally Initiated

C-0-2521 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	1
Maximum value:	2
Default value:	1
Access:	read / write in any Sercos phase

C-0-2522 Oscilloscope Trigger Control Word

This parameter is a 16-bit control word used to enable and trigger a capture of configured signal types. This parameter is activated for both user initiated or internally initiated captures. Configured signal data, for up to 4 signals, are captured and stored in parameters C-0-2511 through C-0-2513 and C-0-2527, respectively. This data can then be uploaded from the control to the relevant oscilloscope signal list parameter. The following figure describes the functions of bits in this parameter:

```
0000000000000111
               |_ Bit 1

Bit 1:  Start recording
Bit 2:  Trigger start
Bit 3:  Enable oscilloscope
```

Fig. 15-12: Bit Description C-0-2522

To initiate a capture manually, bits 1-3 of this parameter are set according to the following table:

C-0-2522 Control	Description
0000000000000000	initial state of oscilloscope
0000000000000100	enable oscilloscope
0000000000000111	enable trigger and start recording
0000000000000100	oscilloscope is enabled and waiting for trigger

Table 15-92: Oscilloscope Control Word

C-0-2522 Attributes

Data length:	2 byte data
Data type:	binary word
Display format:	binary
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0000000000000000
Access:	read / write in any Sercos phase

C-0-2523 Oscilloscope Trigger Status Word

This parameter is a 16-bit status word that displays the current process being performed by parameter *C-0-2522 Oscilloscope Trigger Control Word*. The following figure describes the functions of the bits in this parameter:

```
0000000000001111
               |_ Bit 1


Bit 1: Trigger activated
Bit 2: Data acquisition in progress
Bit 3: System ready (memory allocated)
Bit 4: Capture complete
```

Fig. 15-13: Bit Description C-0-2523

The following table describes the status of this parameter based on the settings in C-0-2522.

C-0-2523 Status	Description
0000000000000000	initial state of oscilloscope
0000000000000100	oscilloscope is ready and waiting for trigger
0000000000000111	data acquisition in progress
0000000000001101	capture complete

Table 15-93: Oscilloscope Control and Status Words

C-0-2523 Attributes

Data length:	2 byte data
Data type:	binary word
Display format:	binary
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	00000000 00000000
Access:	read-only

C-0-2524 Oscilloscope Signal 4 Type

Refer to *C-0-2501 Oscilloscope Signal 1 Type* for a description of this parameter.

C-0-2525 Oscilloscope Signal 4 ID Number

Refer to *C-0-2504 Oscilloscope Signal 1 ID Number* for a description of this parameter.

C-0-2526 Oscilloscope Signal 4 Axis Number

Refer to *C-0-2507 Oscilloscope Signal 1 Axis Number* for a description of this parameter.

C-0-2527 Oscilloscope Signal 4 List

This parameter stores all captured data for the configured signal 4. This information is uploaded from the control after the capture is complete. Refer to *C-0-2511 Oscilloscope Signal 1 List* for the parameters attributes.

Fieldbus/PLC Interface Parameters (C-0-2600 through C-0-2653)

The parameters in this section support the following interfaces:

- Profibus
- DeviceNet
- ControlNet
- Ethernet/IP
- Interbus
- PLC
- PCI Bus

For detailed information on *fieldbus interfaces*, refer to the *Functional Description, sections 7.6, 7.7, and 7.8.*

C-0-2600 Fieldbus/PLC Mapper (cyclic channel) To PLC

This parameter defines the fieldbus object-mapping list transmitted from the PPC-R22.1 to the PLC via the cyclic channel. The parameter can be configured using the Fieldbus Mapper utility. The data is a series of order identifiers of data types.

Format:

```
--CP0.122
 |  || └── Ident. Number
 |  |└──── Data Set
 |  └───── Subclass
 |  └───── Class
 └──────── --  = single item

           00 ┐
           ┆  ├= multiplex item number within a container
           31 ┘
```

Class	Subclass	Data Set	Ident. Number
F (float) I (integer) G (global integer) H (global float)	P	0	variable #
A (axis)	P	1-64 axis #	parameter #
T (task)	P	1-4 task #	parameter #
C (control)	P	0	parameter #
R (register)	X	0	register #

Table 15-94: Mapping List Format

C-0-2600 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	list of configured objects
Access:	read in any Sercos phase / write in Sercos phase 2
Activation:	activates on write to C-0-2601

Note: Data from this parameter is transmitted to the PLC in the order in which it appears.

C-0-2601 Fieldbus/PLC Mapper (cyclic channel) From PLC

This parameter defines the fieldbus object-mapping list transmitted from the PLC to the PPC-R22.1 via he cyclic channel. This parameter can be configured using the Fieldbus Mapper utility. Refer to *C-0-2600 Fieldbus/PLC Mapper (cyclic channel) To PLC* for details.

C-0-2601 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	list of configured objects
Access:	read in any Sercos phase / write in Sercos phase 2

Note: Data from this parameter is transmitted to the PPC-R22.1 control in the order in which it appears.

C-0-2607 Multiplex Control Word

The multiplex control word is used to command the transfer of multiplex data between the PLC and the PPC-R22.1 control.

For detailed information on *multiplex control word*, refer to *Information for the PLC Programmer* in the *Functional Description, sections 7.6, 7.7, and 7.8.*

C-0-2607 Attributes

Data length:	2 bytes
Data type:	Hex display format
Display format:	Hex
Units:	--
Minimum value:	0x0000
Maximum value:	0xFFFF
Default value:	0X0000
Access:	read / write in any Sercos phase

C-0-2608 Multiplex Status Word

The multiplex status word is used to acknowledge the transfer of multiplex data between the PLC and the PPC-R22.1 control.

For detailed information on *multiplex status word*, refer to *Information for the PLC Programmer* in the *Functional Description, sections 7.6, 7.7, and 7.8.*

C-0-2608 Attributes

Data length:	2 bytes
Data type:	Hex display format
Display format:	Hex
Units:	--
Minimum value:	0x0000
Maximum value:	0xFFFF
Default value:	0X0000
Access:	read / write in any Sercos phase

C-0-2611 Fieldbus/PLC Cyclic Channel: Current Number of Misses

This parameter supports the Fieldbus and PCI Bus interfaces.

PCI Bus Support

This parameter displays the current number of missed transfers to/from the cyclic channel. Since the PLC and the control access the DPR completely unsynchronized, it is possible that the control can fail to copy data multiple times. The cyclic channel uses a handshake to ensure that the PLC or the control are not writing and reading from the DPR at the same time. When the control tries to access the DPR, it checks first to see if the handshake bits are set by the PLC. If so, the DPR is not allowed to be accessed. When 10 consecutive missed transfers occur, control parameter C-0-2613, timeout counter, is increment by 1.

PCI Bus and Fieldbus Support

If the control can not copy all the data within one cycle due to resource limitations, the counter is incremented by 1.

C-0-2611 Attributes

Data length:	2 byte data
Data type:	unsigned Integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	10
Default value:	0
Access:	read / write in any Sercos phase

C-0-2612 Fieldbus/PLC Cyclic Channel: Peak Number of Misses

This parameter supports the Fieldbus and PCI Bus interfaces.

PCI Bus Support

This parameter displays the maximum number of missed transfers to/from the cyclic channel. Since the PLC and the control access the DPR completely unsynchronized, it is possible that the control fails multiple times to copy data. It represents the maximum number stored in control parameter C-0-2611.

PCI Bus and Fieldbus Support

If the control can not copy all the data within one cycle due to resource limitations, this counter is incremented by 1.

C-0-2612 Attributes

Data length:	2 byte data
Data type:	unsigned Integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	10
Default value:	0
Access:	read / write in any Sercos phase

C-0-2613 Fieldbus/PLC Cyclic Channel: Timeout Counter

This parameter displays the number of timeouts in the cyclic channel. A timeout is a condition when the control was not able to copy data from/to the cyclic channel for 10 consecutive cycles. Each time that C-0-2611 counts up to 10, this parameter will be incremented by 1. The value will stay at the maximum value if another timeout occurs.

C-0-2613 Attributes

Data length:	2 byte data
Data type:	unsigned Integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	255
Default value:	0
Access:	read / write in any Sercos phase

C-0-2630 Fieldbus Slave Device Address (GPP only)

The hexadecimal value in this parameter identifies the fieldbus slave device address for either Profibus, DeviceNet, ControlNet. The allowable range for fieldbus slave drive address varies based on the configured fieldbus device, as follows:

Fieldbus Interface	Allowable Device Address
Profibus	1-125
DeviceNet	1-63
ControlNet	1-99

Table 15-95: Fieldbus Slave Device Address

C-0-2630 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	hexadecimal
Units:	--
Minimum value:	0x0000
Maximum value:	0x00FF
Default value:	0x003F
Access:	read in any Sercos phase / write in Sercos phase 2
Activation:	activates on write to C-0-2601

C-0-2631 Fieldbus Parameter/PCP Channel Length (GPP only)

When using a Profibus or Interbus fieldbus slave interface, a subset of the cyclic DP (Decentralized Peripheral) channel can be allocated for non-cyclic communications. This subset of the cyclic channel is called the parameter channel for Profibus and the PCP channel for Interbus. This parameter allocates the first 2 to 6 words of the cyclic DP channel as follows:

Channel Length	Profibus (Parameter Channel)	Interbus (PCP Channel)	Description
0x0000	X	X	No parameter channel
0x0004		X	allocates the first 2 words (4 bytes allocated)
0x0008	X		allocates the first 4 words (8 bytes allocated)
0x000C	X		allocates the first 6 words (12 bytes allocated)

Table 15-96: Fieldbus Parameter/PCP Channel Length

C-0-2631 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	hexadecimal
Units:	bytes
Minimum value:	0x0000
Maximum value:	0x000C
Default value:	0x0000
Access:	read in any Sercos phase / write in Sercos phase 2
Activation:	activates on write to C-0-2601

C-0-2632 Fieldbus/PLC Multiplex Method

This parameter sets the primary or secondary multiplex method for fieldbus interfaces. The settings in this parameter vary based on the method configured in Fieldbus Mapper utility.

For detailed information on *multiplexing methods*, refer to the *Functional Description, sections 7.6, 7.7, and 7.8.*

The settings are as follows:

- 0x0000 = primary method
- 0x0001 = secondary method

C-0-2632 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	hexadecimal
Units:	--
Minimum value:	0x0000
Maximum value:	0x0001
Default value:	0x0000
Access:	read in any Sercos phase / write in Sercos phase 2
Activation:	activates on write to C-0-2601

C-0-2633 Fieldbus Baud Rate (DeviceNet only) (GPP only)

This parameter displays the configured baud rate in hexadecimal for the installed DeviceNet slave card. The baud rate is set using the Fieldbus Mapper utility when configuring a DeviceNet slave card. The allowable baud rates are:

- 0x0001E848 = 125 KBaud
- 0x0003D090 = 250 KBaud
- 0x0007A120 = 500 KBaud

C-0-2633 Attributes

Data length:	4 byte data
Data type:	unsigned integer
Display format:	hexadecimal
Units:	--
Minimum value:	0x00000000
Maximum value:	0x0007A121
Default value:	0x0001E848
Access:	read in any Sercos phase / write in Sercos phase 2
Activation:	activates on write to C-0-2601

C-0-2635 Fieldbus/PLC Error Reaction

This parameter determines how the control will react to a fieldbus error. Fieldbus error reaction can be configured using the Fieldbus Mapper Utility. The valid error reaction settings are as follows:

- 0x0000 = shutdown (default)
- 0x0001 = warning
- 0x0002 = ignore

C-0-2635 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	hexadecimal
Units:	--
Minimum value:	0x0000
Maximum value:	0x0002
Default value:	0x0000
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-2636 Fieldbus/PLC Word Swap

This parameter determines the order in which fieldbus slave interface or PCI-bus interface non-cyclic and/or cyclic data are transmitted between VisualMotion controls and an external PLC. The lower byte of this parameter is used for fieldbus slave interface, while the upper byte is used for the PCI-bus interface.

The following format is used:

Format:

```
0x0101
  └┘└┘
  │  └──── Fieldbus Slave Interface (lower byte)
  │        0x0000 = no swapping
  │        0x0001 = 32-bit word swapping
  │        0x0002 = explicit message byte swapping
  │        0x0003 = both 32-bit and explicit
  │
  └─────── PCI-bus Interface (upper byte) GMP only
           0x0000 = no swapping
           0x0100 = word swapping
           0x0200 = byte swapping
           0x0300 = both word and byte
```

The following table shows the different swapping types available:

Type	Word 1	Word 2
No swapping (original order of data)	byte 1 byte 2	byte 3 byte 4
32-bit word swapping for fieldbus & Word swapping for PCI-bus	byte 3 byte 4	byte 1 byte 2
Explicit message byte swapping for fieldbus & Byte swapping for PCI-bus	byte 2 byte 1	byte 4 byte 3
Both 32-bit and explicit & Both word and byte	byte 4 byte 3	byte 2 byte 1

Table 15-97: Word and Byte Swapping

32-bit Object Word Swapping (Fieldbus Slave) and Word Swapping (PCI-bus)

The setting of this option determines the order in which the two data words in any 32-bit (double word) cyclic or non-cyclic mapped object are transmitted.

Explicit Message Byte Swapping (Fieldbus Slave)

The setting of this option determines the order in which the bytes of non-cyclic data >4 bytes long are transmitted.

Byte Swapping (PCI-bus)

The setting of this option determines the order in which the bytes of any cyclic mapped object are transmitted.

C-0-2636 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	hexadecimal
Units:	--
Minimum value:	0x0000
Maximum value:	0x0303
Default value:	0x0000
Access:	read in any Sercos phase / write in Sercos phase 2
Activation:	activates on write to C-0-2601

C-0-2637 Fieldbus/PLC Slave Firmware Version

This parameter displays the current firmware version and release date of the installed and configured fieldbus or PLC interface. If no fieldbus or PLC interface is detected, this parameter contains no value.

Format:

```
Firmware version  Date released  Driver version
V01.034           08.10.99       V04T17__24022003
```

C-0-2637 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	0
Maximum value:	39 characters
Default value:	firmware version of installed fieldbus interface
Access:	read-only

C-0-2638 Fieldbus/PLC Available Cyclic IN Parameters

This parameter contains a list of allowable control, axis and task parameters that can be used as cyclic input data for parameter C-0-2600 Fieldbus/PLC Mapper (cyclic channel) To PLC.

Format:

C-0-0002
A-0-0020
T-0-0002

C-0-2638 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	list of supported parameters
Access:	read-only

C-0-2639 Fieldbus/PLC Available Cyclic OUT Parameters

This parameter contains a list of allowable control, axis and task parameters that can be used as cyclic output data for parameter C-0-2601 Fieldbus/PLC Mapper (cyclic channel) From PLC.

Format:

C-0-0002

C-0-2639 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	list of supported parameters
Access:	read-only

C-0-2640 PLC Connection Options (GMP only)

This parameter is used to establish communication between the PLC and PPC-P11.1 control. If set to 0, the PLC interface will not be initialized in the firmware and no communication will be established. If bit 1 (the least significant bit) is set to 1 (default) the interface will be initialized and communication will take place. Cycle power to the control for the change to take effect.

C-0-2640 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00000010
Default value:	00000000 00000001
Access:	read in any Sercos phase / write in Sercos phase 2
Activation:	requires power cycle

C-0-2641 PLC Input Register List

This parameter contains the configured list of registers that are placed in the register channel of the Dual Port RAM for transfer to the PLC. The register channel is used to cyclically transfer register data between the VisualMotion controls and a PLC at the current Sercos cycle. The register channel is limited to 128 registers (256 bytes) in one direction (input or output).

The list can be configured using the *Parameter Overview* as follows:

1. Select ***Data ⇒ Parameters***.
2. Double click on parameter C-0-2641 to open the parameter list edit window.
3. Right click and select **Insert Item (INS)**.
4. Enter the register number to add it to the list (e.g. 1 for reg. 1).
5. Click on **OK** to return to the *Parameter Overview* window.

C-0-2641 Attributes

Data length:	variable length 2 byte data
Data type:	integer array
Display format:	unsigned integer
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-2642 PLC Output Register List

This parameter contains the configured list of registers that are retrieved from the register channel of the Dual Port RAM. Refer to *C-0-2641 PLC Input Register List* for a description of the register channel.

Note: It is not advisable to list status registers in this parameter. These registers are read-only and no data will be transmitted across the DPR memory.

C-0-2642 Attributes

Data length:	variable length 2 byte data
Data type:	integer array
Display format:	unsigned integer
Units:	--
Minimum value:	0
Maximum value:	128
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-2643 PLC Lifecounter Check: Number of Retries (GMP only)

VisualMotion PLC and controls check the life counter every Sercos cycle in the I/O task. If the value is the same as the previous cycle, the PPC will recheck the life counter before issuing an error, if C-0-2643 is non-zero. The value is a multiple of retries.

Currently, the PLC interface checks every Sercos cycle with a retry rate fixed in firmware to 10ms, giving a life count timeout of 20-40ms.

C-0-2643 Attributes

Data length:	2 byte data
Data type:	unsigned Integer
Display format:	unsigned Integer
Units:	--
Minimum value:	0
Maximum value:	255
Default value:	40
Access:	read / write in any Sercos phase

C-0-2644 PLC Lifecounter Check: Current Number of Misses (GMP only)

The PLC life counter is checked every Sercos cycle. If the value has not changed since the last Sercos cycle, this parameter is incremented by 1. The count runs from 0 to the maximum value set in C-0-2643 (PLC life counter check: # of Retries). When the life counter changes between two consecutive Sercos cycles, this parameter is reset to 0. This parameter is used for PCI-bus and PLC interfaces.

C-0-2644 Attributes

Data length:	2 byte data
Data type:	unsigned Integer
Display format:	unsigned Integer
Units:	--
Minimum value:	0
Maximum value:	65535
Default value:	0
Access:	read / write in any Sercos phase

C-0-2645 PLC Lifecounter Check: Peak Number of Misses (GMP only)

This parameter monitors C-0-2644 (PLC life counter check: Current # of misses), peak count between 0 and the value set in C-0-2643 (PLC life counter check: # of Retries), and holds that value until a larger value is encountered. This parameter is only used for the motion / logic interface.

C-0-2645 Attributes

Data length:	2 byte data
Data type:	unsigned Integer
Display format:	unsigned Integer
Units:	--
Minimum value:	0
Maximum value:	65535
Default value:	0
Access:	read / write in any Sercos phase

C-0-2646 PLC Lifecounter Check: Number of Timeouts (GMP only)

This counter increments by one every time C-0-2645 (PLC life counter check: Peak # of misses) encounters the number of missed life counter updates specified in C-0-2643 (PLC life counter check: # of Retries). A count incremented of one represents a PLC life counter update failure and processed by the control according to the selected error reaction specified in C-0-2635. This parameter is used for PCI-bus and motion / logic interfaces.

C-0-2646 Attributes

Data length:	2 byte data
Data type:	unsigned Integer
Display format:	unsigned Integer
Units:	--
Minimum value:	0
Maximum value:	65535
Default value:	0
Access:	read / write in any Sercos phase

C-0-2651 PLC Register Channel: Current Number of Misses (GMP only)

This parameter displays the current number of missed transfers to/from the register channel. The register channel uses a handshake to ensure that either the PLC or the control are not writing and reading from the DPR at the same time. When the PPC tries to access the DPR, it checks first to see if the handshake bits are set by the PLC. If so, it is not allowed to access the DPR. Since the PLC and the control access the DPR completely unsynchronized, it is possible that the control fails multiple times to copy data. In addition, if the control firmware can not copy all the data within one cycle due to resource limitations, the counter is increased. This parameter is used for PCI-bus and motion / logic interfaces.

C-0-2651 Attributes

Data length:	2 byte data
Data type:	unsigned Integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	10
Default value:	0
Access:	read / write in any Sercos phase

C-0-2652 PLC Register Channel: Peak Number of Misses (GMP only)

This parameter displays the maximum number of missed transfers to/from the register channel. It represents the maximum number stored in control C-0-2651. The register channel uses a handshake to ensure that either the PLC or the control are not writing and reading from the DPR at the same time. When the control tries to access the DPR, it checks first to see if the handshake bits are set by the PLC. If so, it is not allowed to access the DPR. Since the PLC and the control access the DPR completely unsynchronized, it is possible that the control fails multiple times to copy data. In addition, if the control firmware can not copy all the data within one cycle due to resource limitations, the counter is increased. This parameter is used for PCI-bus and motion / logic interfaces.

C-0-2652 Attributes

Data length:	2 byte data
Data type:	unsigned Integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	10
Default value:	0
Access:	read / write in any Sercos phase

C-0-2653 PLC Register Channel: Timeout Counter (GMP only)

This parameter displays the number of timeouts in the register channel. A timeout is a condition when the control was not able to copy data from/to the register channel for 10 consecutive cycles. Each time that C-0-2651 counts up to 10, this parameter will be incremented by 1. The value will stay at the maximum value if another timeout occurs. This parameter is used for PCI-bus and motion / logic interfaces.

C-0-2653 Attributes

Data length:	2 byte data
Data type:	unsigned Integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	255
Default value:	0
Access:	read / write in any Sercos phase

Encoder Interface Card Parameters (C-0-2800 through C-0-2878)

C-0-2800 MEC Hardware Version

This parameter displays the MEC hardware version. For example, "0x3003", the version information is read from left-to-right as follows:

0x3003

- EPLD revision
- EPLD version
- Hardware version
- Hardware type

C-0-2800 Attributes

Data length:	2 bytes
Data type:	Hex display format
Display format:	Hex
Units:	--
Minimum value:	0x0000
Maximum value:	0xFFFF
Default value:	0X0000
Access:	read-only

C-0-2801 MEC Firmware Version

This parameter displays the MEC's application firmware version along with its compilation date and time (if applicable).

C-0-2801 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	read by control
Access:	read-only

C-0-2802 MEC Configuration

This parameter allows users to enable or disable the two encoder inputs on the LAG master encoder card. This parameter has no affect when a master encoder card is not present.

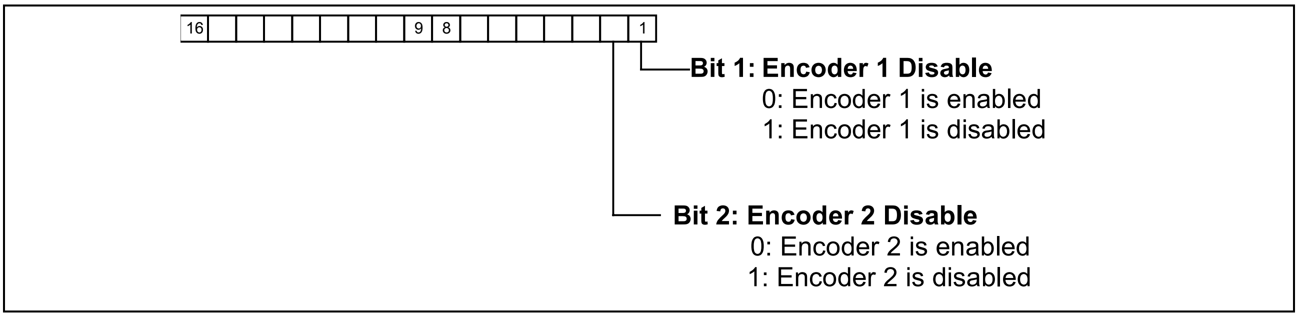


Fig. 15-14: MEC Configuration

Encoder 2 Disable (Bit 2)	Encoder 1 Disable (Bit 1)	Hardware Configuration (from C-0-2810)	Function
0	0	2 x EnDat (0x2)	Encoder 1 and encoder 2 are enabled
0	1	2 x EnDat (0x2)	Encoder 1 is disabled and encoder 2 is enabled
1	0	2 x EnDat (0x2)	Encoder 1 is enabled and encoder 2 is disabled
1	1	2 x EnDat (0x2)	All MEC functions are disabled

Table 15-98: Functionality Configuration

C-0-2802 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00000011
Default value:	00000000 00000000
Access:	read / write in Sercos phase 2

C-0-2805 MEC Error Type

This parameter displays the MEC's current error type as follows:

Value	Description
0	no error
1	encoder 1 error
2	encoder 2 error
3	Encoder Interface system error

Table 15-99: MEC Error Type

C-0-2805 Attributes

Data length:	2 byte data
Data type:	unsigned short
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	3
Default value:	0
Access:	read-only

C-0-2806 MEC Error Group

This parameter displays the MEC's current error group used for internal purposes.

C-0-2806 Attributes

Data length:	2 byte data
Data type:	unsigned short
Display format:	unsigned decimal
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0
Access:	read-only

C-0-2807 MEC Error Number

This parameter displays the MEC's current error number used for internal purposes.

C-0-2807 Attributes

Data length:	2 byte data
Data type:	unsigned short
Display format:	unsigned decimal
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0
Access:	read-only

C-0-2810 MEC Error Detection

This parameter displays the encoder configuration as detected by the MEC.

Value	Description
0x0	No PC104 card installed
0x2	2 x EnDat / 1 V p-p

Table 15-100: MEC Error Detection

C-0-2810 Attributes

Data length:	2 bytes
Data type:	Hex display format
Display format:	Hex
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0X0000
Access:	read-only

C-0-2811 MEC Error Type Detection

This parameter displays the encoder type as detected by the MEC.

Value	Description
0x00	unknown
0x01	encoder 1 is a single-turn
0x10	encoder 2 is a single-turn
0x02	encoder 1 is a multi-turn
0x20	encoder 2 is a multi-turn
0x04	encoder 1 is a 1 V p-p Sinusoidal
0x40	encoder 2 is a 1 V p-p Sinusoidal

Table 15-101: MEC Error Type Detection

C-0-2811 Attributes

Data length:	2 bytes
Data type:	Hex display format
Display format:	Hex
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0X0000
Access:	read-only

C-0-2815 MEC Direction of Encoder 1

This parameter identifies the direction polarity of the connected encoder.

Bit 4 Value	Description
0	positive polarity
1	negative polarity

Table 15-102: MEC Direction of Encoder 1

C-0-2815 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00001000
Default value:	00000000 00000000
Access:	read / write in Sercos phase 2

C-0-2816 MEC Direction of Encoder 2

This parameter identifies the direction polarity of the connected encoder.

Bit 4 Value	Description
0	positive polarity
1	negative polarity

Table 15-103: MEC Direction of Encoder 2

C-0-2816 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00001000
Default value:	00000000 00000000
Access:	read / write in Sercos phase 2

C-0-2817 MEC Encoder 1 Resolution

This parameter is set to zero (default) when using an EnDat encoder for encoder 1. When using a 1Vpp Sinusoidal encoder, set this parameter to the resolution (non-zero) of the 1-Volt peak-to-peak sinusoidal encoder.

Note: Incorrect configuration of this parameter will result in encoder errors or incorrect position values.

C-0-2817 Attributes

Data length:	4 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	0
Maximum value:	65535
Default value:	0
Access:	read in any phase / write in Sercos phase 2

C-0-2818 MEC Encoder 2 Resolution

This parameter is set to zero (default) when using an EnDat encoder for encoder 2. When using a 1Vpp Sinusoidal encoder, set this parameter to the resolution (non-zero) of the 1-Volt peak-to-peak sinusoidal encoder.

Note: Incorrect configuration of this parameter will result in encoder errors or incorrect position values.

C-0-2818 Attributes

Data length:	4 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	0
Maximum value:	65535
Default value:	0
Access:	read in any phase / write in Sercos phase 2

Option Card PLS Interface Parameters (C-0-2901 through C-0-2943)

C-0-2901 PLS1 Start Output Register

This parameter defines the start (first) output register assigned to the Option Card PLS's output modules. Two consecutive registers are used to display the status of the Option Card PLS outputs. The bits of the displayed register number represent outputs 1-16, respectively. The bits of the next register represent optional outputs 17-32. These registers are updated every Sercos cycle.

C-0-2901 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	1
Maximum value:	1024
Default value:	70
Access:	read / write in any Sercos phase

C-0-2902 PLS1 Start Mask Register

This parameter defines the start (first) mask register assigned to the start output register in parameter C-0-2901. The next consecutive mask registers is assigned the next consecutive output register. The bits of the displayed register number mask outputs 1-16, respectively. The bits of the next register, mask outputs 17-32.

C-0-2902 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	1
Maximum value:	1024
Default value:	74
Access:	read / write in any Sercos phase

C-0-2903 PLS1 Build Command

While operating in Sercos phase 4, the user can modify existing Option Card PLS parameters (for example, change settings for on / "off" positions). This parameter is used to issue a build command for the Option Card PLS table, based on the new settings. The new settings are calculated by the control and a new PLS table is generated. The new settings are not effective immediately and require an activation command from parameter C-0-2905. The status of the build command is monitored in parameter C-0-2904.

Bit 4	Bit 3	Bit 2	Bit 1	Description
0	0	0	0	No command to build PLS
0	0	1	1	Command to build PLS

Table 15-104: PLS1 Build Command

Note: This build command is not necessary if modifications to the Option Card PLS are made while in parameter mode (P2). The PLS table is calculated and activated every time a Sercos phase transition from Phase 2 to 3 is performed.

C-0-2903 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00000011
Default value:	00000000 00000000
Access:	read / write in any Sercos phase

C-0-2904 PLS1 Build Status

This parameter displays the status of a build command issued in C-0-2903. The status indications are described in the following table:

Bit 4	Bit 3	Bit 2	Bit 1	Description
0	0	0	0	No command to build PLS
0	0	1	1	Command to build PLS was successful
0	1	1	1	Command to build PLS is processing.
1	1	1	1	Communication error, build command was not successful

Table 15-105: PLS1 Build Status

C-0-2904 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00001111
Default value:	00000000 00000000
Access:	read-only

C-0-2905 PLS1 Activate Command

This parameter is used to activate the Option Card PLS table built with C-0-2903. If this command is issued before the build command C-0-2903 is done, an error is issued.

C-0-2905 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00000011
Default value:	00000000 00000000
Access:	read / write in any Sercos phase

C-0-2906 PLS1 Activate Status

This parameter displays the status of a activate command issued in C-0-2905. If the activate command had been executed successfully, this parameter displays a binary 3. If an error is encountered, a binary 15 will be displayed. A binary 7, indicates that the control is still processing the command.

C-0-2906 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00001111
Default value:	00000000 00000000
Access:	read-only

C-0-2907 PLS1 Error Code

This parameter displays an internal diagnostic error code for the Option Card PLS that is used for internal purposes.

C-0-2907 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	1
Maximum value:	512
Default value:	0
Access:	read-only

C-0-2908 PLS1 Extended Error Code

This parameter displays an extended error code for the error displayed in C-0-2907 PLS1 Error Code.

C-0-2908 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	1
Maximum value:	512
Default value:	0
Access:	read-only

C-0-2909 PLS1 Hardware ID

This parameter displays the hardware ID for the installed Option Card PLS. The hardware ID is displays as a hexadecimal value. The high byte represents the hardware ID for the Option Card PLS. The lower byte represents the hardware revision for the Option Card PLS.

C-0-2909 Attributes

Data length:	2 byte data
Data type:	unsigned decimal
Display format:	hexadecimal
Units:	--
Minimum value:	0x0000
Maximum value:	0xFFFF
Default value:	0x0000
Access:	read-only

C-0-2910 PLS1 Software ID

This parameter displays the firmware version installed on the Option Card PLS. Option Card PLS firmware is included in the firmware.

C-0-2910 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

C-0-2920 PLS1 Switch On List

This parameter displays a list of "on" positions of all 96 Option Card PLS switches. The units for the value are based on the assigned master (degrees, mm, or inch). If the switch's on and "off" position have the same value, the output will not turn on. Any modifications to "on" positions require a build (C-0-2903) and activate (C-0-2905) command.

C-0-2920 Attributes

Data length:	variable length 4 byte data
Data type:	float array
Display format:	float
Units:	--
Minimum value:	0
Maximum value:	128 list elements
Default value:	0
Access:	read / write in any Sercos phase
Activation:	activates after entering and exiting Sercos phase 2 ***or*** after building (C-0-2903) and activating (C-0-2905) PLS

C-0-2921 PLS1 Switch Off List

This parameter displays a list of the "off" positions of all 96 Option Card PLS switches. The units for the value are based on the assigned master (degrees, mm, or inch). Any modifications to "off" positions require a build (C-0-2903) and activate (C-0-2905) command.

C-0-2921 Attributes

Data length:	variable length 4 byte data
Data type:	float array
Display format:	float
Units:	--
Minimum value:	0
Maximum value:	128 list elements
Default value:	0
Access:	read / write in any Sercos phase
Activation:	activates after entering and exiting Sercos phase 2 ***or*** after building (C-0-2903) and activating (C-0-2905) PLS

C-0-2922 PLS1 Switch Output List

This parameter displays a list of the Option Card PLS's output assignments for all 96 switches. If a switch is not used, the value is 0. Otherwise, the value indicates the output number (1-32). Any modifications to the switch output list requires a build (C-0-2903) and activate (C-0-2905) command.

C-0-2922 Attributes

Data length:	variable length 2 byte data
Data type:	integer
Display format:	unsigned integer
Units:	--
Minimum value:	0
Maximum value:	128 list elements
Default value:	0
Access:	read / write in any Sercos phase
Activation:	activates after entering and exiting Sercos phase 2 ***or*** after building (C-0-2903) and activating (C-0-2905) PLS

C-0-2930 PLS1 Output Master List

This parameter displays a list of 32 Option Card PLS outputs. The value displayed for each output represents the number of the master assigned to that output. An Option Card PLS can have up to 8 masters of type ELS Master, ELS Group, and drive.

C-0-2930 Attributes

Data length:	variable length 4 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	0
Maximum value:	32 list elements
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-2931 PLS1 Output Lead Time List

This parameter displays a list of 32 Option Card PLS outputs. The value displayed for each output represents the lead time, in µs, assigned to each output. Lead time is the amount of time that the output is enabled prior to reaching the switch's "on" position. This value is used by the control to calculate a position based on the switch's "on" position and the current speed of the PLS master. Any modifications to the output lead time list requires a build (C-0-2903) and activate (C-0-2905) command.

The assigned lead time is always active, regardless of any additional functions, such as a one shot. Lead times are written in increments of 250 µs. The allowable range is 0 - 500000 µs (for a total of 200 ms). When using the PLS tool, the user will be prompt when entering a value that is not in increments of 250 µs.

Note: There are no individual lead times for switches. All switches assign to an output will use the output's lead time.

C-0-2931 Attributes

Data length:	variable length 4 byte data
Data type:	float array
Display format:	float
Units:	µs
Minimum value:	0
Maximum value:	32 list elements
Default value:	0
Access:	read / write in any Sercos phase
Activation:	activates after entering and exiting Sercos phase 2 ***or*** after building (C-0-2903) and activating (C-0-2905) PLS

C-0-2932 PLS1 Output Lag Time List

This parameter displays a list of 32 Option Card PLS outputs. The value displayed for each output represents the lag time, in µs, assigned to each output. Lag time is the amount of time that the output is disabled prior to reaching the switch's "off" position. This value is used by the control to calculate a position based on the switch's "off" position and the current speed of the PLS master. This parameter is only available when C-0-2934 PLS1 Output Mode List is set to 0. Any modifications to the output lag time list requires a build (C-0-2903) and activate (C-0-2905) command.

The assigned lag time is always active, regardless of any additional functions, such as a one shot. Lag times are written in increments of 250 µs. The allowable range is 0 - 500000 µs (for a total of 200 ms). When using the PLS tool, the user will be prompt when entering a value that is not in increments of 250 µs.

Note: There is no individual lag times for switches. All switches assign to an output will use the output's lag time.

C-0-2932 Attributes

Data length:	variable length 4 byte data
Data type:	float array
Display format:	float
Units:	--
Minimum value:	0
Maximum value:	32 list elements
Default value:	0
Access:	read / write in any Sercos phase
Activation:	activates after entering and exiting Sercos phase 2 ***or*** after building (C-0-2903) and activating (C-0-2905) PLS

C-0-2933 PLS1 Output One Shot List

This parameter displays a list of 32 Option Card PLS outputs. The value displayed for each output represents the on time (µs) assigned to that output, when configured as a one shot output. On times for one shots are written in increments of 250 µs. The allowable range is 0-1000000 µs (for a total of 1 s). When using the PLS tool, the user will be prompt when entering a value that is not in increments of 250 µs. Any modifications to the output one shot list requires a build (C-0-2903) and activate (C-0-2905) command.

Note: There are no individual one shot times for switches. All switches assign to an output will use the output's one shot time.

C-0-2933 Attributes

Data length:	variable length 4 byte data
Data type:	float array
Display format:	float
Units:	--
Minimum value:	0
Maximum value:	32 list elements
Default value:	0
Access:	read / write in any Sercos phase
Activation:	activates after entering and exiting Sercos phase 2 ***or*** after building (C-0-2903) and activating (C-0-2905) PLS

C-0-2934 PLS1 Output Mode List

This parameter displays a list of 32 Option Card PLS outputs. The value displayed for each output represents the Mode assigned to that output. The mode determines which functions can be assigned to an output. Any modifications to the output mode list requires a build (C-0-2903) and activate (C-0-2905) command.

The available mode setting are:

- 0 - output lead and lag time are available.
- 1 - output lead and one shot are available, no lag time.

C-0-2934 Attributes

Data length:	variable length 2 byte data
Data type:	integer array
Display format:	unsigned integer
Units:	--
Minimum value:	0
Maximum value:	32 list elements
Default value:	0
Access:	read / write in any Sercos phase
Activation:	activates after entering and exiting Sercos phase 2 ***or*** after building (C-0-2903) and activating (C-0-2905) PLS

C-0-2935 PLS1 Output Direction List

This parameter displays a list of 32 Option Card PLS outputs. The value displayed for each output represents the direction in which each switch's On and "off" position will be detected for the assigned output. If the outputs are only active in one direction, the outputs will be turned off if the direction changes, independent of the previous state of the output. If the outputs are controlled via a oneshot and the direction is changed, the outputs will be on for the specified oneshot time. However, during the time where the direction has changed, but the oneshot is still on, the oneshot is not retriggerable. Any modifications to the output direction list requires a build (C-0-2903) and activate (C-0-2905) command.

The allowable direction values within the list are:

- 0 - Positive
- 1 - Negative
- 2 - Positive and Negative

C-0-2935 Attributes

Data length:	variable length 2 byte data
Data type:	integer array
Display format:	integer
Units:	--
Minimum value:	0
Maximum value:	32 list elements
Default value:	0
Access:	read / write in any Sercos phase
Activation:	activates after entering and exiting Sercos phase 2 ***or*** after building (C-0-2903) and activating (C-0-2905) PLS

C-0-2936 PLS1 Output Hysteresis List

This parameter displays a list of 32 Option Card PLS outputs. The value displayed for each output represents the hysteresis assigned to that output.

C-0-2936 Attributes

Data length:	variable length 4 byte data
Data type:	float array
Display format:	float
Units:	--
Minimum value:	0
Maximum value:	32 list elements
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-2940 PLS1 Master Type List

This parameter displays a list of 8 Option Card PLS masters. The value displayed for each master represents the input type assigned to that PLS master. The available master input types are as follows:

- 0 – not assigned
- 1 – ELS Master
- 2 – ELS Group
- 3 – Drive (primary or secondary encoder)
- 4 - Used for internal purposes
- 5 - Used for internal purposes

C-0-2940 Attributes

Data length:	variable length 2 byte data
Data type:	integer array
Display format:	integer
Units:	--
Minimum value:	0
Maximum value:	8 list elements
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-2941 PLS1 Master Number List

This parameter displays a list of 8 Option Card PLS masters. The value displayed for each master represents the number that is assigned to the corresponding input master type in C-0-2940. The available input type numbers for the master types assigned in C-0-2940 are as follows:

- 1-6 for master type 1 (ELS Master)
- 1-8 for master type 2 (ELS Group)
- 1-64 for master type 3 (Drive)

Note: The minimum, maximum and default values listed in the attributes table is the allowable range for the data contained in the list and not for the number of items (elements) contained in the list.

C-0-2941 Attributes

Data length:	variable length 2 byte data
Data type:	integer array
Display format:	unsigned integer
Units:	--
Minimum value:	0
Maximum value:	8 list elements
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-2942 PLS1 Master Encoder List

This parameter displays a list of 8 Option Card PLS masters. The value displayed for each master represents the encoder type for the master type (Drive) assigned in C-0-2940. The available encoder types are as follows:

- 1 – primary encoder
- 2 – secondary encoder

Note: The minimum, maximum and default values listed in the attributes table is the allowable range for the data contained in the list and not for the number of items (elements) contained in the list.

C-0-2942 Attributes

Data length:	variable length 2 byte data
Data type:	integer array
Display format:	unsigned integer
Units:	--
Minimum value:	0
Maximum value:	8 list elements
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-2943 PLS1 Master Phase Offset List

This parameter displays a list of 8 Option Card PLS masters. The value displayed for each master represents the relative Sercos phase offset assigned to that PLS master. The offsets can be set in any Phase and have an immediate effect. No build or activation is required.

Note: The minimum, maximum and default values listed in the attributes table is the allowable range for the data contained in the list and not for the number of items (elements) contained in the list.

C-0-2943 Attributes

Data length:	variable length 4 byte data
Data type:	float array
Display format:	float
Units:	--
Minimum value:	0
Maximum value:	8 list elements
Default value:	0
Access:	read / write in any Sercos phase

CAM Table Parameters (C-0-3100 through C-0-3140)

C-0-3100 CAM Tags

This parameter is a list of name(s) given to each configured and downloaded CAM table to the control. Each CAM name can be up to 20 characters in length. The order in which the names are displayed identifies each CAM in ascending order, starting with 1.

C-0-3100 Attributes

Data length:	variable length 1 bytes data
Data type:	string array
Display format:	text
Units:	--
Minimum value:	0
Maximum value:	40
Default value:	0
Access:	read / write in any Sercos phase

Note: This parameter can be modified in any Phase as long as the CAM is not active in the program.

C-0-3101 CAM Table 1 through C-0-3137 CAM Table 37

Each parameter is a list that contains all the points of a built CAM table for CAM's 1-37. CAMs are built and transferred in VisualMotion under menu select ***Tools*** ⇒ ***CAM Builder*** or via an icon program.

C-0-3101 Attributes

Data length:	variable length 1 bytes data
Data type:	string array
Display format:	text
Units:	--
Minimum value:	0
Maximum value:	1025 points
Default value:	0
Access:	read / write in any Sercos phase

Note: This parameter can be modified in any Phase as long as the CAM is not active in the program.

C-0-3138 CAM Table 38 through C-0-3140 CAM Table 40

Each parameter is a list that contains all the points of a built CAM table for CAM's 38-40. These CAMs are pre-defined for the ELS lock on / lock off feature.

For detailed information on *Electronic Line Shafting*, refer to the *Functional Description, section 6.1.*

C-0-3138 through C-0-03140 Attributes

Data length:	variable length 1 bytes data
Data type:	string array
Display format:	text
Units:	--
Minimum value:	0
Maximum value:	1025 points
Default value:	0
Access:	read / write in any Sercos phase

Note: This parameter can be modified in any Phase as long as the CAM is not active in the program.

C-0-3141 CAM Type

The CAM type configuration word is used to set the CAM format and enable the non-zero starting point used for all CAMs.

Bit 1: CAM Type

This bit stores the current CAM type used in the system. This parameter is written at program activation. The following table lists the supported CAM types:

Bit 1	CAM Type
0	Degree format
1	Percentage format

Table 15-106: CAM Type

Bit 2: Enable Non Zero Starting Point

The difference between CAM point 1 and CAM point 1024 is calculated. If the difference is less than 50% (modulo/2), then point 1025 = point 1. Otherwise, point 1025 = point 1 + 100 % (modulo) * NumberModulo (+/- number of modulos of movement for the CAM).

Bit 2	Description
0	Last point is either equal to 0% or 100%
1	Last point = first point or first point + 100%

Table 15-107: Enable Non Zero Starting Point

Note: Writing to this parameter builds the default cams 38-40. If these cams are in use, the error message "Cannot store CAM: already active for axis 0" is displayed.

C-0-3141 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00000011
Default value:	00000000 00000001
Access:	read in any Sercos phase / write in Sercos phase 2

Position Monitoring Group Parameters

The position monitoring group functionality in allows the system to monitor the deviation between a primary slave position and up to 5 slave axes. Up to 8 groups can be commissioned, each containing 6 parameters. The following parameters are used to store the pertinent data from the Position Monitoring Group tool.

For detailed information on *Position Monitoring Group*, refer to the *Functional Description, section 5.3*.

C-0-32x1 PMG # Maximum Allowed Deviation Window

The value in this parameter is scanned each time a group's enable bit (Reg. 86 bits 1-8) is set. It represents the maximum positional deviation allowed in a grouping of axes. The two methods used for positional deviation monitoring are as follows:

- *Deviation from Primary Signal* – deviation between the primary signal and slave axes.
- *Min/Max Group Deviation Window* - deviation between a minimum and maximum axes positions in a group.

The following table lists the 8 similar parameters in the PMG functionality used to store the maximum deviation window.

Parameter	Description
C-0-3201	PMG 1 Maximum Allowed Deviation Window
C-0-3211	PMG 2 Maximum Allowed Deviation Window
C-0-3221	PMG 3 Maximum Allowed Deviation Window
C-0-3231	PMG 4 Maximum Allowed Deviation Window
C-0-3241	PMG 5 Maximum Allowed Deviation Window
C-0-3251	PMG 6 Maximum Allowed Deviation Window
C-0-3261	PMG 7 Maximum Allowed Deviation Window
C-0-3271	PMG 8 Maximum Allowed Deviation Window

Table 15-108: Maximum Allowed Deviation Window Parameters

Note: Changes to the maximum deviation window, while a group is enabled, has no immediate effect. The group's control register enable bit (Reg. 86 bits 1-8) must be disabled and then set before a deviation window change can take effect.

C-0-32x1 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005
Minimum value:	+0.0
Maximum value:	+90.0
Default value:	+0.0
Access:	read / write in any Sercos phase

C-0-32x2 PMG # List of Axes

This list parameter displays the axes contained in a Position Monitoring Group. A total of 6 axes can be configured in a PMG. The first axis in the list is the primary slave followed by up to 5 slave axes. The following table lists the 8 similar parameters in the PMG functionality used to store the List of Axes per group.

Parameter	Description
C-0-3202	PMG 1 List of Axes
C-0-3212	PMG 2 List of Axes
C-0-3222	PMG 3 List of Axes
C-0-3232	PMG 4 List of Axes
C-0-3242	PMG 5 List of Axes
C-0-3252	PMG 6 List of Axes
C-0-3262	PMG 7 List of Axes
C-0-3272	PMG 8 List of Axes

Table 15-109: List of Axes Parameters

C-0-32x2 Attributes

Data length:	variable length 2 bytes data
Data type:	integer array
Display format:	unsigned integer
Units:	--
Minimum value:	0
Maximum value:	6 list elements
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

C-0-32x3 PMG # List of Position Offsets

This list parameter displays the offsets for each axis contained in the PMG group. All axes in control parameter C-0-32x2 PMG # List of Axes must be defined in this list (e.g., the item count of both lists must match). If the list is empty then the default value of 0.0 is initialized for each axis in the group. The offset value is added to the position feedback (A-0-0102) value. This combined value is then used for deviation detection. The following table lists the 8 similar parameters in the PMG functionality used to store the List of Position Offsets per group.

Parameter	Description
C-0-3203	PMG 1 List of Position Offsets
C-0-3213	PMG 2 List of Position Offsets
C-0-3223	PMG 3 List of Position Offsets
C-0-3233	PMG 4 List of Position Offsets
C-0-3243	PMG 5 List of Position Offsets
C-0-3253	PMG 6 List of Position Offsets
C-0-3263	PMG 7 List of Position Offsets
C-0-3273	PMG 8 List of Position Offsets

Table 15-110: List of Position Offsets Parameters

C-0-32x3 Attributes

Data length:	variable length 4 bytes data
Data type:	float array
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	6 list elements
Default value:	0
Access:	read / write in any Sercos phase

C-0-32x4 PMG # Current Peak Group Deviation

This parameter displays the current peak position deviation between the furthermost axis position in the group and the primary slave. Every Sercos cycle, the system recalculates this value and stores it in this parameter. If a deviation error is detected, the system no longer updates this parameter and stores the maximum deviation position. The following table lists the 8 similar parameters in the PMG functionality used to store the Current Peak Group Deviation per group.

Parameter	Description
C-0-3204	PMG 1 Current Peak Group Deviation
C-0-3214	PMG 2 Current Peak Group Deviation
C-0-3224	PMG 3 Current Peak Group Deviation
C-0-3234	PMG 4 Current Peak Group Deviation
C-0-3244	PMG 5 Current Peak Group Deviation
C-0-3254	PMG 6 Current Peak Group Deviation
C-0-3264	PMG 7 Current Peak Group Deviation
C-0-3274	PMG 8 Current Peak Group Deviation

Table 15-111: Current Peak Group Deviation Parameters

C-0-32x4 Attributes

Data length:	4 byte
Data type:	float number
Display format:	signed decimal
Units:	A-0-0005 of primary slave
Minimum value:	0.0
Maximum value:	current peak group deviation
Default value:	current peak group deviation
Access:	read-only

C-0-32x5 PMG # Maximum Deviation

This parameter displays the peak position deviation encountered between the furthermost axis position in a group from the primary slave after it was enabled. Every Sercos cycle, the system checks the current value in C-0-32x4 PMG # Current Peak Group Deviation. If the value of C-0-32x4 is greater, than the new value is stored this parameter. If a deviation error is detected, the system no longer updates this parameter and stores the maximum deviation position.

Note: Writing a 0 to this parameter will reset the value and display the new current maximum deviation in the group.

The following table lists the 8 similar parameters in the PMG functionality used to store the Maximum Deviation per group.

Parameter	Description
C-0-3205	PMG 1 Maximum Deviation
C-0-3215	PMG 2 Maximum Deviation
C-0-3225	PMG 3 Maximum Deviation
C-0-3235	PMG 4 Maximum Deviation
C-0-3245	PMG 5 Maximum Deviation
C-0-3255	PMG 6 Maximum Deviation
C-0-3265	PMG 7 Maximum Deviation
C-0-3275	PMG 8 Maximum Deviation

Table 15-112: Maximum Deviation Parameters

C-0-32x5 Attributes

Data length:	4 byte
Data type:	float number
Display format:	signed decimal
Units:	A-0-0005 of primary slave
Minimum value:	0.0
Maximum value:	current maximum group deviation
Default value:	current maximum group deviation
Access:	read / write in any Sercos phase

C-0-32x6 PMG # Configuration

This parameter displays the current Position Monitoring Group configuration. This following figure describes the bit functions used in this binary parameter.

```
0000000000000001
               |_ Bit 1

Bit 1:     Deviation Method
           0=min/max slave, 1=primary signal

Bit 2:     Error Reaction
           0=warning, 1=fatal error (Stop)

           Primary Source
Bits 5-3:  000 = A-0-0102 (Fdbk)
           001 = A-0-0101 (Pos Cmd)
           010 = ELS System Group Position
           110 = ELS Group Output
           100 = A-0-0195 (Fdbk Data #1 from A-0-0185)
           101 = A-0-0196 (Fdbk Data #2 from A-0-0186)

Bit 6:     Suppress PMG Warnings
           0=default, 1=suppress warnings

Bits 7-16: reserved
```

Fig. 15-15: PMG # Configuration Bit Description

The following table lists the 8 similar parameters in the PMG functionality used to store the PMG Configuration per group.

Parameter	Description
C-0-3206	PMG 1 Configuration
C-0-3216	PMG 2 Configuration
C-0-3226	PMG 3 Configuration
C-0-3236	PMG 4 Configuration
C-0-3246	PMG 5 Configuration
C-0-3256	PMG 6 Configuration
C-0-3266	PMG 7 Configuration
C-0-3276	PMG 8 Configuration

Table 15-113: PMG Configuration Parameters

C-0-32x6 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000001 00000000
Default value:	00000000 00000000
Access:	read in any Sercos phase / write in Sercos phase 2

15.7 Task Parameters - Class T

Each user task (A-D) has a set of parameters that selects options and displays status information. This following section describes these parameters for tasks and coordinated motion.

Task Setup (T-0-0001 and T-0-0002)

T-0-0001 Task Motion Type

This parameter is set automatically by the user program and is set to 1 when coordinated motion is used. Until a valid program is activated, all tasks are non-coordinated by default. The following values define the task motion type as follows:

- 0 = no coordinated motion
- 1 = coordinated motion

T-0-0001 Attributes

Data length:	4 byte data
Data type:	integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	1
Default value:	0
Access:	read-only

T-0-0002 Task Options

This parameter specifies an automatic start option for the task and how the task will react to errors.

```
0000000000000001
               |_ Bit 1

Bit 1:  Run task during error
Bit 2:  Task errors do not shutdown other
        task
Bit 3:  Parameter transfer errors are
        warning
Bit 4:  Automatically start tasks
Bit 5:  Ignore task and drive errors
        from other tasks
```

Fig. 15-16: Bit Description T-0-0002

Bit 1: Run Task during Errors

The setting in this bit determines whether or not this task will run during errors. The following setting are available:

0 = Shutdown task on errors (default)

1 = Run task during errors

For detailed information on *Error Reaction*, refer to volume 1 of the *Rexroth VisualMotion 11 Functional Description*, *chapter 11*.

Bit 2: Task Errors Do Not Shutdown Other Tasks

This bit is used to determine if task errors that occur in this task should shutdown other tasks. The following setting are available:

0 = Shutdown other tasks on task error (default)

1 = Task errors do not shutdown other tasks

For detailed information on *Error Reaction*, refer to volume 1 of the *Rexroth VisualMotion 11 Functional Description*, *chapter 11*.

Bit 3: Parameter Transfer Errors are Warnings

This bit is used to determine if a parameter transfer error should be processed as an error or a warning. The following setting are available:

0 = Parameter transfer errors can shutdown task (default)

1 = Parameter transfer errors are warnings

Setting this bit to 1 allows the current task to continue running if a parameter transfer error occurs. The message "*205 Parameter Transfer Warning: see Task diag.*" is issued. If a parameter value is critical to the operation of the task, the task error bit (bit 5 of task status register) can be tested after the parameter transfer, or set this bit to 0.

For detailed information on *task registers 22-25*, refer to *chapter 16*.

Bit 4: Automatically Start Tasks

This option automatically starts tasks and keeps them running. All task control register bits are ignored. The task is switched to automatic mode and started when exiting parameter mode or when clearing an error. The task is stopped when parameter mode is selected. This option can be used for supervisory or communications tasks, or to allow the system to start at power-up without any operator intervention.

The Override_Auto_Start bit (bit 2 in the task control register) can be used to temporarily disable this function.

For detailed information on *task registers 2-5*, refer to *chapter 16*.

Note:	The delay time for task configured as auto-start actually begins upon entering Sercos phase 4. The delay time may vary depending on the quantity of axes associated with the task.

Bit 5: Ignore Task and Drive Errors from Other Tasks

This bit is used to determine whether or not this task will ignore task and drive errors from other tasks. The following settings are available:

0 = Do not ignore task and drive errors from other tasks (default)

1 = Ignore task and drive errors from other tasks

For detailed information on *Error Reaction*, refer to volume 1 of the *Rexroth VisualMotion 11 Functional Description*, *chapter 11*.

T-0-0002 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	0000000000000000
Maximum value:	--
Default value:	0000000000000000
Access:	read in any Sercos phase / write in Sercos phase 2

T-0-0003 Task Instruction Rate

The processing time among programming task instructions (icons) are equally divided. This parameter specifies the number of instruction commands (icons) that will be processed in this task before moving to the next task.

T-0-0003 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	1
Maximum value:	16
Default value:	8
Access:	read in any Sercos phase / write in Sercos phase 2

Coordinated Motion (T-0-0005 through T-0-0026)

T-0-0005 World Position Units

This parameter selects the display units for position, speed, and acceleration data for coordinated motion. No unit conversions are performed when changing this parameter. The following settings are available:

- 0 = inches
- 1 = millimeters
- 2 = radians

T-0-0005 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	2
Default value:	1
Access:	read in any Sercos phase / write in Sercos phase 2

T-0-0010 Kinematic Number

The kinematic number represents a library routine that identifies the kinematic to be used for coordinated motion. Kinematic routines are application specific and unique to hardware configurations.

T-0-0010 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	0
Maximum value:	17
Default value:	0
Access:	read-only

T-0-0011 Coordinated X Axis

This parameter sets the axis number corresponding to the x-axis of this task. An axis number is assigned under ***Setup*** ⇒ ***Axis***. If a 0 is set in this parameter, no coordinated x-axis is assigned to this task.

T-0-0011 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	8
Default value:	0
Access:	read-only

T-0-0012 Coordinated Y Axis

This parameter sets the axis number corresponding to the y-axis of this task. An axis number is assigned under ***Setup*** ⇒ ***Axis***. If a 0 is set in this parameter, no coordinated y-axis is assigned to this task.

T-0-0012 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	8
Default value:	0
Access:	read-only

T-0-0013 Coordinated Z Axis

This parameter sets the axis number corresponding to the z-axis of this task. An axis number is assigned under ***Setup*** ⇒ ***Axis***. If a 0 is set in this parameter, no coordinated z-axis is assigned to this task.

T-0-0013 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	8
Default value:	0
Access:	read-only

T-0-0020 Maximum Path Speed

This parameter sets the maximum path speed allowed for all coordinated motion axes assigned to this task. The speed entry in the absolute and relative point tables is a percentage of this maximum path speed.

The maximum path speed can also be limited by the maximum velocity parameter (A-0-0020) of each axis and each drive's bipolar velocity limit parameter (S-0-0091).

T-0-0020 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	inches/min or mm/min
Minimum value:	+0.001
Maximum value:	1e+07
Default value:	+1000.0
Access:	read / write in any Sercos phase

T-0-0021 Maximum Acceleration

This parameter sets the maximum acceleration allowed for all coordinated motion axes assigned to this task. The acceleration entry in the absolute and relative point tables is a percentage of this maximum acceleration.

The maximum acceleration can also be limited by the maximum acceleration parameter (A-0-0021) of each axis.

T-0-0021 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	inches/sec^2 or mm/sec^2
Minimum value:	+0.01
Maximum value:	1e+07
Default value:	+200.0
Access:	read / write in any Sercos phase

T-0-0022 Maximum Deceleration

This parameter sets the maximum deceleration allowed for all coordinated motion axes assigned to this task. The deceleration entry in the absolute and relative point tables is a percentage of this maximum deceleration.

T-0-0022 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	T-0-0005 / sec²
Minimum value:	+0.01
Maximum value:	1e+07
Default value:	+200.0
Access:	read / write in any Sercos phase

T-0-0023 Look Ahead Distance

This parameter sets the minimum look ahead distance that the control's path planner uses to calculate a path. The system will never have a smaller pre-calculated distance in the system queue than the value in this parameter until the path is completed. The Look Ahead Distance is a mechanism used for preventing the path motion instructions from reading more points from the point table.

Setting the look ahead distance to a large value can improve the overall system performance. The length of the Look Ahead Distance determines how many path segments that are to be processed by the path planner. Of course, if the parameter is set too large, the path planner may process many statements before physical motion takes place. For segments that are queued up, the path planner cannot identify potential real-time actions that may stop motion or require a program branch and the action may not happen with as fast of a response.

Decreasing the Look Ahead Distance value causes the path planner to process fewer coordinated motion program statements ahead of the current commanded position. This results in a lower potential for wasted calculations in specific cases in which the path may be aborted earlier than expected.

There is a minimum of two geometry segments that are automatically placed in the queue. Even if the Look Ahead Distance is set to 0, there will be at least the next two unprocessed segments regardless of the Look Ahead Distance value. If the segments are blended together then there is a minimum of three segments on the system queue.

In order to provide a smooth motion for a dynamic path the Look Ahead Distance should be set to the size of the smallest segment. When dynamically adding segments to the path, the decision of adding another segment must be finalized when the current position is at least two segments before the newer segments that are to be added. Blending between segments can increase the number of segments in the path planner to be processed. Generally, the Look Ahead Distance should be twice the length of the longest blend distance. If all blend distances are zero, the look ahead distance should be equal to the shortest geometry segment.

T-0-0023 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	T-0-0005
Minimum value:	+0.0
Maximum value:	+10000.0
Default value:	+10.0
Access:	read / write in any Sercos phase

T-0-0024 Velocity Override

The velocity override provides a method to equally slow all motion in a task. When a velocity override factor is specified for coordinated motion, all velocities in the point table are multiplied by this factor as they are used. When a velocity override is specified for non-coordinated motion that is generated by the digital drive, each velocity command is multiplied by this factor before the command is executed.

T-0-0024 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	percentage
Minimum value:	+0.099
Maximum value:	+100.0
Default value:	+100.0
Access:	read / write in any Sercos phase

T-0-0025 Maximum Jog Increment

This parameter defines the maximum distance that is used for incremental coordinated jogging. C-0-0042 World Large Increment and C-0-0043 World Small Increment percent parameters are based on this value.

T-0-0025 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	T-0-0005
Minimum value:	+0.0001
Maximum value:	+1000.0
Default value:	+1.0
Access:	read / write in any Sercos phase

T-0-0026 Maximum Jog Velocity

This parameter defines the maximum velocity used for coordinated jogging. The world fast (C-0-0045) and world slow (C-0-0046) percent parameters are based on this value.

T-0-0026 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	T-0-0005 / min
Minimum value:	+0.0001
Maximum value:	+100000.0
Default value:	+100.0
Access:	read / write in any Sercos phase

T-0-0027 Path Smoothing Filter Constant

This parameter defines the filter constant used to smooth the path profile with coordinated moves and coordinated jogging. A percentage of this filter constant value can be adjusted in the point table under the % Jerk column for user programs. This percent value will be read from the point table only at the end of a geometry segment that is not blended when the path velocity is at zero at the time the icon program executes a coordinated motion move icon. If this is not the case, the previously read percent value will remain active. When the control is jogging it will use this value for the path profile.

T-0-0027 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	ms
Minimum value:	+0.0
Maximum value:	+200.0
Default value:	+40.0
Access:	read / write in any Sercos phase

T-0-0050 Kinematic Value 1 through T-0-0059 Kinematic Value 10

Each segment in a robotic arm is represented as a length using parameters T-0-0050 through T-0-0059. These parameters have default values which are dependant on the currently active Kinematic. They are set to their default values upon program activation (either through program management, project synchronization or via power cycle).

For detailed information on *Coordinated Motion,* refer to the *Functional Description, section 6.2*.

T-0-0050 through T-0-0059 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	--
Minimum value:	+0.0
Maximum value:	+10.0
Default value:	dependent on Kinematic
Access:	read in any Sercos phase / write in Sercos phase 2

Coordinated Motion Status (T-0-0100 through T-0-0113)

Coordinated motion statuses are read-only dynamically updated parameters that provide status values for each task.

T-0-0100 Target Point Number

This parameter displays the current target point.

For example:

If 10 is displayed here, motion to point ABS [10] or REL [10] is taking place, or the machine is currently at this point.

CAUTION

The point displayed in T-0-0100 is the target point set in the path planner but necessarily the actual position. Depending on value in parameter T-0-0023 Look Ahead Distance, the actual position of the drive might well be at the previous segment moving towards the next.

T-0-0100 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0 (current target point in path planner)
Access:	read-only

T-0-0101 Segment Status

This parameter displays the status of the current segment. The segment status codes are listed in the table below. The codes are valid for the current segment, except for codes 0 and 1 that are transitional or do not apply to the current segment. Use code 6 to check if motion is halted due to a path/stop. The following codes are available:

Code	Segment Status	Activity
0	Segment ready	Path Planner is currently not running
1	Acceleration	Acceleration in progress
2	Constant velocity	Constant velocity in progress (at target velocity)
3	Blending	Blending in progress
4	Target deceleration	Deceleration to target position in progress
5	Controlled stop	Controlled stop taking place (error, jog or path/stop)
6	Stopped	Motion has stopped (error, jog or path/stop)
7	Segment at target	Jerk filter active
8	Segment done	Segment completed

Table 15-114: Task Segment Status Bits

T-0-0101 Attributes

Data length:	4 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0 (dynamically updated by control)
Access:	read-only

T-0-0111 Current X Position

This parameter displays the current commanded position, in world coordinates, for the x-axis set in T-0-0011.

T-0-0111 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	T-0-0005
Minimum value:	--
Maximum value:	--
Default value:	+0.0
Access:	read-only

T-0-0112 Current Y Position

This parameter displays the current commanded position, in world coordinates, for the y-axis set in T-0-0012.

T-0-0112 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	T-0-0005
Minimum value:	--
Maximum value:	--
Default value:	+0.0
Access:	read-only

T-0-0113 Current Z Position

This parameter displays the current commanded position, in world coordinates, for the z-axis set in T-0-0013.

T-0-0113 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	T-0-0005
Minimum value:	--
Maximum value:	--
Default value:	+0.0
Access:	read-only

Task Status (T-0-0120 through T-0-0200)

T-0-0120 Task Operating Mode

This parameter displays the current operating mode of the task. Information about the task's state, etc., is available in the control and status I/O registers. The following settings are available:

- 0 = initialization
- 1 = parameter
- 2 = manual
- 3 = automatic

T-0-0120 Attributes

Data length:	4 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	0
Maximum value:	3
Default value:	read by control
Access:	read-only

T-0-0122 Task Diagnostic Message

This parameter displays the current diagnostic message for the task. During normal operation, a **Msg1** icon in the user program sets this message. If an error occurs during task execution, this diagnostic message is overwritten with an error message.

T-0-0122 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	read by control
Access:	read-only

T-0-0123 Task Status Message

This parameter displays the current status message for this task. A **Msg1** icon in the user program sets this message and aids the operator when debugging. This message is not overwritten with an error message, allowing debugging of an error condition set in the Task Diagnostic Message. The following strings are available:

- Initialization Mode
- Parameter Mode
- Manual Mode
- Automatic Mode

T-0-0123 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	read by control
Access:	read-only

T-0-0130 Current Instruction Pointer

This parameter returns a hexadecimal value equal to the current task's execution address (i.e. the instruction pointer). The hex value is an offset from the start of the program.

For example:

"0x000000F0" indicates that the program counter is at 0xF0, or 240 bytes from the start of the program.

T-0-0130 Attributes

Data length:	4 byte data
Data type:	unsigned integer
Display format:	hexadecimal
Units:	--
Minimum value:	0x00000000
Maximum value:	0xFFFFFFFF
Default value:	read by control
Access:	read-only

T-0-0131 Current Instruction

This parameter displays the current instruction pointer and mnemonic of the current instruction. The mnemonic is in the base code format generated by the control's compiler. This parameter is used for debugging and troubleshooting programs.

Format:

```
0028 START
 |      |_______ current mnemonic
 |_____________ current instruction pointer
```

T-0-0131 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	read by control
Access:	read-only

T-0-0132 Instruction Pointer at Error

This status parameter displays the instruction pointer where the last task error occurred. This parameter is used for debugging and troubleshooting programs.

T-0-0132 Attributes

Data length:	4 byte data
Data type:	unsigned integer
Display format:	hexadecimal
Units:	--
Minimum value:	0x00000000
Maximum value:	0xFFFFFFFF
Default value:	read by control
Access:	read-only

T-0-0133 Composite Instruction Pointer

This parameter dynamically displays a flag and a current instruction pointer indicating the relative memory address where a program instruction is executed. This parameter is used by VisualMotion Toolkit to determine which icon to highlight in the program flow when using *Show Program Flow*.

Note: The highlighting of icons in nested subroutines, when using *Show Program Flow*, is limited to 10 levels.

The first number (flag) identifies where the instruction resides in the program as well as how many pointers identify the relative memory address where the instruction is being executed. The numbers, in hexadecimal format that follow the flag are the relative memory address where the instruction is being executed.

Flag (First Number)	Description
1	Instruction is located in the main task
2 to 11	Instruction is located in a subroutine
(-2) to (-11)	Instruction is in an event

Table 15-115: Composite Instruction Pointer

T-0-0133 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	read by control
Access:	read-only

T-0-0135 Current Subroutine

This parameter indicates the current subroutine being executed with the function number and name. This includes tasks, non-accessible functions, functions, and subroutines. If function number and name information is not included in the user program file, the string “NONE” is returned.

T-0-0135 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	read by control
Access:	read-only

T-0-0136 Stack Variable Data

This parameter displays the current stack variable data for function arguments and local variables defined in the **Start** icon. Stack variables are valid only while the program flow is within a task, subroutine, or event function. The maximum number of stack variables is 16. If there are no defined arguments or local variables in a task, subroutine, or event function, this parameter displays "NONE".

T-0-0136 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	read by control
Access:	read-only

T-0-0137 Subroutine Breakpoint

This task parameter specifies the index number of the subroutine to which program execution will halt when task breakpoint is enabled. To enable Task breakpoint set bit 11 of the Task_Control register to (1). Task program flow continues with a (0-1) transition of bit 6 (Cycle_Start) in the Task_Control register.

T-0-0137 Attributes

Data length:	4 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	0
Maximum value:	65535
Default value:	0
Access:	read / write in any Sercos phase

T-0-0138 Sequencer Information

This task parameter displays information about the currently running sequencer in both index and name format. The index indicates the current row in the sequence list or step list. If no sequencer is running, this parameter displays "NONE".

T-0-0138 Attributes

Data length:	variable length 1 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	NONE
Access:	read-only

T-0-0200 Last Active Event Number

This parameter displays the index of the current or last active event in the event (EVT) table. The value can be used to access other information (message, status, function, etc.) contained in the event table.

T-0-0200 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0
Access:	read-only

Task Parameter Lists (T-0-2000 and T-0-2001)

T-0-2000 List of All Parameters

This parameter contains a list of all task parameters that are part of the current firmware version.

T-0-2000 Attributes

Data length:	variable length 4 byte data
Data type:	unsigned integer
Display format:	IDN (parameter number)
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

T-0-2001 List of Required Parameters

This parameter contains a list of all required task parameters that are part of the archive / restore function.

T-0-2001 Attributes

Data length:	variable length 4 byte data
Data type:	unsigned integer
Display format:	IDN (parameter number)
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

15.8 Axis Parameters – Class A

Axis parameters are used to configure the axis and provide limits for coordinated motion. Some parameters only apply to a specific axis mode (coordinated or single-axis).

Axis Setup (A-0-0001 through A-0-0038)

A-0-0001 Task Assignment

This parameter associates an axis with a user task or IndraLogic. The assignment of a task or PLC to an axis is done under ***Setup* ⇒ *Axes***. The following settings are available:

0 = No task selected	1 = Task A	2 = Task B
3 = Task C	4 = Task D	10 = PLC

A-0-0001 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	10
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

A-0-0002 Type of Positioning

This parameter defines the type of positioning for an axis as either normal positioning or lagless positioning. This is relevant for an axis configured as single-axis, coordinated, ratio, ELS Sercos phase synch, control CAM or drive CAM. Lagless positioning helps reduce the following error during the acceleration or deceleration of an axis. The following settings are available:

0 = Normal positioning

1 = Lagless positioning

A-0-0002 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	1
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

A-0-0003 Axis Motion Type

This parameter defines the operating mode of the axis and configures data in the AT and MDT Sercos telegrams. Operating mode are define when setting up an axis under ***Setup*** ⇒ ***Axes***.

Motion Type	Description
0 = Disabled	no active motion type
1 = Single Axis	The axis moves independently of other axis using a target position with defined velocity and acceleration / deceleration ramps.
2 = Coordinated Axis	The path planner in the control calculates the position of axis to produce a coordinated move using predefined ABS or REL point tables and kinematic.
3 = Velocity Mode	This motion type operates the axis at constant velocity without a position control loop. The control sends only velocity commands to the digital drive. The digital drive maintains the velocity profile.
4 = Ratio Slave	This motion type designates the axis as a slave to the master axis designated in Parameter A-0-0030 Ratio Mode Master Axis.
5 = ELS Slave	This motion type designates the axis as an Electronic Line Shafting (ELS) slave.
6 = Torque Mode	This motion type designates the axis the run in torque mode, with torque commands sent from the control and no velocity or position loop.
7 = Control CAM Axis	This motion type designates the axis to run from a CAM table stored on the control and uses an ELS Virtual or Real Master.
8 = Torque Following Mode	This motion type designates the axis to run with **-torque following mode.
9 = Coordinated Articulation	This motion type designates the axis as a control CAM with coordinate transformation.

Table 15-116: Axis Motion Type

A-0-0003 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	9
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

A-0-0004 Axis Options

This parameters sets all the necessary options for a Rexroth digital drive.

```
0000000000000001
               |_ Bit 1

Bit 1:   Position initialization
Bit 2:   Positioning mode
Bit 3&4: ELS synchronization mode
Bit 5:   Enable Measuring Wheel Services
Bit 6:   Optional cyclic data
Bit 7:   Enable Redundant Motor Encoder
         (IndraDrive MPx-04 only)
Bit 8:   Drive PLS Fast Write (DKC only)
Bit 9:   ELS secondary mode
Bit 10:  Disable ELS shortest path
Bit 11:  Positioning using secondary encoder
Bit 12:  Drive disable method
Bit 13:  Linear axis modulo positioning
Bit 14:  Configure minimum cyclic data
Bit 15:  Disable modulo positioning for rotary axis
Bit 16:  Disable automatic scaling
```

Fig. 15-17: Bit Description A-0-0004

Bit 1: Position Initialization

This option is used for non-absolute measurement systems, such as single-turn or incremental encoder. During Sercos phase 2 initialization, the feedback position value can remain unchanged (bit 1 = 0) or be initialized with the "Starting Position Value" in P-0-0019 (bit 1 = 1).

0 = Keep feedback value in Phase 2 (default)

1 = Reset feedback in Phase 2

Keep feedback value in Phase 2

When set to 0, a different starting position value can be entered through the Sercos service channel or the feedback value will be reset to the default one-revolution measurement.

Reset Feedback in Phase 2

When set to 1, the control overwrites the feedback position A-0-0102 with the value in P-0-0019 Starting Position Value after the system is switch out of parameter mode.

Bit 2: Positioning Mode

0 = Linear Positioning Mode (default)

1 = Rotary Positioning Mode

The positioning mode is valid for all axis types. All relevant scaling parameters are automatically set in the drive. If an axis is an ELS slave, its mode is automatically set to rotary positioning.

When rotary positioning mode is enabled in the drive, position is in degrees, velocity in RPM, and acceleration in radians/sec. Single-axis and velocity mode values are entered in these units. Events for single-axis and velocity mode work with both rotary and linear positioning.

Coordinated motion and events are compatible with rotary positioning only if motion takes place within the modulo value and does not rollover. If an axis is coordinated, velocity is in linear units/sec and acceleration in units/sec.

Linear Positioning Mode

When set to 0, linear positioning is selected. The units and scaling are specified with parameter A-0-0005 Linear Position Units. Absolute positioning is enabled in the drive if bit 13 (Linear Axis Modulo Positioning) is set 0. Setting bit 13 to 1 enables linear axis using modulo position.

Rotary Positioning Mode

When set to 1, rotary positioning is enabled. The position, velocity, and acceleration units and scaling are fixed at the drive. Modulo positioning is enabled by default, with a rollover value specified in drive parameter S-0-0103. Setting bit 15 (Disable Modulo Positioning for Rotary Axis) to 1 disables modulo positioning and sets absolute positioning.

Bits 4 and 3: ELS Synchronization Mode

This option sets the type of synchronization for an electronic line shafting axis. These bits are set automatically at program activation by the ELS/INIT instruction.

Bit 4	Bit 3	Mode
0	0	**Velocity Synchronization**. The axis runs in velocity mode, with its velocity equal to the master's velocity times the translation ratio. The ratio can be fine adjusted at run time.
0	1	**Phase Synchronization**. The axis runs in the operating mode selected in parameter A-0-0002 Type of Positioning, and maintains a Sercos phase relationship according to the Sercos phase offset and translation ratio parameters. The Sercos phase offset can be adjusted at run time.
1	1	**CAM Table**. The axis is linked to a CAM and synchronized to a master. Position relationship is maintained according to the CAM table, ratio and stretch factors. Position offset can be adjusted at run time.

Table 15-117: ELS Synchronization Mode Bits

Bit 5: Enable Measuring Wheel Services

When set to 1, bit 5 enables measuring wheel services. The AT is configured as follows:

- S-0-0386 (active feedback) is written to the AT
- S-0-0051 (feedback 1) and S-0-0052 (feedback 2) are not written to the AT

Drive parameter S-0-0386 provides the control with an active position feedback value. When using the measuring wheel services (bit 5), all feedback position values (S-0-0386) are used by the control for all internal functions (jog mode, stop icon, etc.), instead of S-0-0051 or S-0-0053.

Note: When using Measuring Wheel Services (Bit 5=1), Bit 11 should be set to 0. Measuring wheel services is not possible together with Positioning Using Secondary Encoder (Bit 11=1).

Parameter S-0-0386 is supported by the following drive firmware:

DIAX04 (ELS05V(>31)) and ECODRIVE03 (SGP03V22 and SGP20V11)

Bit 6: Optional Cyclic Data

0 = Use smallest cyclic data configuration (default)

1 = Include optional cyclic data (velocity feedback, programmed acceleration)

Use Smallest Cyclic Data Configuration

When set to 0, the smallest amount of cyclic data is used to increase the number of drives on the ring and increase cycle time. Velocity feedback is normally read through the service channel, which can take up to 50ms using a program command.

Include Optional Cyclic Data

To include velocity feedback in the cyclic data for all axis modes, set this bit to 1. In single-axis mode, programmed acceleration is also sent cyclically instead of through the service channel. For IndraDrive drives, S-0-0359 *Positioning Deceleration* is added to the MDT.

Up to five optional, IDNs can be specified with parameters A-0-0180 Optional Command ID #1 through A-0-0186 Optional Feedback ID #2 in addition to or exclusive of this option bit.

Bit 7: Enable Redundant Motor Encoder (IndraDrive MPx 04 only)

When Axis Options parameter 4 (A-x-0004), bit 7 is set to a 1, the GPP firmware will set the IndraDrive parameter, P-0-0185, Control word of encoder 2 (optional encoder), bit 3 equal to 1.

Note: A-0-0004, bit 7 (Redundant Encoder) and bit 11 (Positioning Using Secondary Encoder) cannot be enable at the same time.

Bit 8: Drive PLS Fast Write (DKC2.3 only)

0 = Drive PLS Fast Write disabled (default)

1 = Drive PLS Fast Write enabled

When set to 1, the Drive PLS fast write is enabled. Enabling this feature will force the Sercos multiplex channel on. The PLS fast write applies when using the Calc Drive PLS icon.

Bit 9: ELS Secondary Mode

0 = Secondary mode is single axis mode (default)

1 = Secondary mode is velocity mode

For an ELS axis, this bit selects the secondary, non-synchronized mode for a digital drive. This is the default mode until the **ELSmode** icon switches the axis into ELS velocity or Sercos phase synchronous mode.

Secondary Mode is Single Axis Mode

When set to 0, the default secondary mode is set to single axis positioning mode. Velocity mode may also be selected when using the **ELSmode** icon, if it is configured in the cyclic data using parameters A-0-0180 Optional Command ID #1 to A-0-0196.

Secondary Mode is Velocity Mode

When set to 1, the secondary mode is set to velocity mode. On drives that don’t support single axis mode, the secondary mode is always velocity, regardless of this bit.

Bit 10: Disable ELS Shortest Path

0 = Shortest path positioning for ELS Sercos phase adjust (default)

1 = Shortest path disabled

This bit selects the positioning method used for the ELS Sercos phase adjust move.

Shortest Path Positioning for ELS Phase Adjust

When set to 0, the axis takes the shortest path within a revolution. If the difference in position is between 180 and 360 degrees, the axis travels counterclockwise.

Shortest Path Disabled

When set to 1, shortest path is disabled. A move within one revolution travels in the positive direction if the programmed position is positive, and in the negative if, it is negative.

Bit 11: Positioning Using Secondary Encoder

0 = Use primary feedback (encoder 1) (default)

1 = Use secondary feedback (encoder 2)

This bit configures the digital drive to use Encoder 2 to close the position loop and provide cyclic feedback from drive parameter S-0-0053. This option must be set if the drive is using linear motor firmware.

Bit 12: Drive Disable Method

0 = Stop axis immediately (default)

1 = Coast to a stop

This bit changes the response of the drive after a fatal error or when the disable bit in the axis control register is set. It configures the way the control sets the enable bits (14 and 15) in the Master control word S-0-0134.

Stop Axis Immediately

When set to 0, the drive immediately commands the motor to zero velocity before disabling torque and applying the brake. This option should be used for coordinated motion or linear motion, where coasting can cause damage or injury.

Coast to a Stop

When set to 1, the drive immediately disables torque, causing the motor to coast, stopping with its own inertia. This should be used in some types of line shafting applications, where immediate disabling of the drive could cause damage.

Bit 13: Linear Axis Modulo Positioning

0 = Linear axis uses absolute positioning (default)

1 = Linear axis uses modulo positioning

Linear Axis Uses Absolute Positioning

When set to 0, absolute motion is enabled, with signed positions and no modulo. This option should be used for coordinated motion and most absolute positioning applications.

Linear Axis Uses modulo Positioning

When set to 1, the modulo value in S-0-0103 is used. When the axis position reaches the modulo value, it resets to 0. There are no negative position values. Unlike rotary mode, the scaling of position, velocity, and acceleration is linear (inches or mm). This option should be used for continuous indexing operations.

Bit 14: Configure Minimum Cyclic Data

0 = Default or maximum cyclic data (default)

1 = Minimum cyclic data

ELS or CAM Axes

Default or Maximum Cyclic Data

When set to 0, parameter S-0-0036 (velocity command value) is included in the cyclic data, which allows real-time update of velocity when the axis is in its secondary mode (A-0-0004, bit 9=1).

Minimum Cyclic Data

When set to 1, parameter S-0-0036 (velocity command value) is removed from the cyclic data. Now that the velocity feedback value is read through the service channel only, no ramping or real-time control can be done on the axis. Therefore, the axis should only be run in ELS synchronization mode.

Single-axis mode

Default or Maximum Cyclic Data
When set to 0, parameter S-0-0259 (position velocity value) is included in the cyclic data for applications where the velocity is changed often in the user program.

Minimum Cyclic Data
When set to 1, parameter S-0-0259 (position velocity value) is removed from the cyclic data to allow the maximum number of drives and options and the minimum Sercos cycle time. The velocity is then read through the service channel. This option is not recommended in applications where the velocity is changed often in the user program.

Velocity mode

Minimum Cyclic Data
When set to 1, parameter S-0-0182 Manufacturer Class 3 Diagnostics is removed from the AT. This is required when using a REFU drive, since they do not support S-0-0182.

Ratio Axis Master

Default or Maximum Cyclic Data
When set to 0, parameter S-0-0040 (velocity feedback value) is included in the cyclic data for applications where the velocity is changed often in the user program.

Minimum Cyclic Data
When set to 1, parameter S-0-0040 (velocity feedback value) is removed from the cyclic data.

Bit 15: Disable Modulo Positioning for Rotary Axis

0 = Absolute positioning (default)

1 = Modulo positioning

Absolute Positioning
When set to 0 (default), the rotary axis position resets to the modulo value for every revolution.

Modulo Positioning
When set to 1, modulo positioning is disabled for a rotary axis. Positions less than zero are negative, and the modulo parameter is not used. This should be used only with positioning applications, not with indexing, ELS, or continuous motion.

Bit 16: Disable Automatic Scaling

0 = Control sets Sercos scaling parameters (RECOMMENDED) (default)

1 = Control does not set Sercos scaling parameters

For most drives, the control sets Sercos scaling parameters such as S-0-0044 (velocity data scaling type) and S-0-0076 (position data scaling type). If a drive does not accept the control's parameter settings, these parameters can be set manually.

Note: This bit should only be set at the recommendation of the Bosch Rexroth applications department.

A-0-0004 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	displays current axis options
Access:	read in any Sercos phase / write in Sercos phase 2

A-0-0005 Linear Position Units

This parameter sets the units of measurement that will be used for linear axis positioning, speed, and acceleration data. No unit conversions are performed when changing this parameter to a different unit of measure. The following units are available:

0 = inches

1 = millimeters (default)

2 = degrees

The drive's display and scaling parameters are automatically set by the control. All data is transmitted in decimal format with the same resolution as the data transmitted by the drive. Velocity, acceleration, and jerk data are always transmitted with and accuracy of three-decimal places. The following decimal places are used for position data:

Units	Decimal Places	Maximum Value
inches	5	21474.83648
millimeters	4	214748.3648
radians	6	2147.483648

Table 15-118: Decimal Accuracy for A-0-0005

If an axis is in rotary mode, the drive automatically sets the position to degrees, the velocity to RPM, and the acceleration to rads/sec/sec. In rotary mode, the scaling and display units in this parameter do not apply.

Note: Changing this parameter affects the scaling of drive parameters. If you need to retain such constants, you must save each required drive mechanical parameter before modifying this parameter.

A-0-0005 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	inches, millimeters or degrees
Minimum value:	0
Maximum value:	2
Default value:	1
Access:	read in any Sercos phase / write in Sercos phase 2

A-0-0006 Reference Options

This parameter selects options for reference position monitoring and homing.

```
0000000000000001
               |_ Bit 1

Bit 1:    Issue error when drive is enabled
Bit 16-2: Reserved for future development
```

Fig. 15-18: Bit Description A-0-0006

Bit 1: Issue Error when Drive is Enabled

When set to 1, the control immediately issues the error “500 Axis D is not referenced” if the drive is enabled with no referenced position. The control reads drive parameter S-0-0403, Position Feedback Status, to determine if the position is referenced.

While the drive is disabled, the Set Absolute Measurement command (P-0-0012) is used to set the reference position. This option should only be set when an absolute encoder is used. It prevents an incremental homing procedure from being used.

A-0-0006 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	00000000 00000000
Access:	read in any Sercos phase / write in Sercos phase 2

A-0-0007 Configuration Mode

This parameter allows drives to be excluded from the user program and initialized to single-axis or velocity mode. An error will not be issued if the drive is not found on the Sercos ring. The following settings are available:

Setting	Description
0 (default)	the axis can be used in the program with its defined axis type and its presence on the ring will be verified
1	the drive is excluded from the program, but can be jogged using default values
2	the drive is disabled and is put into a torque-free mode after its initialized

Table 15-119: Settings for A-0-0007

A-0-0007 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	2
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

A-0-0009 Drive PLS Register

This parameter displays the register assigned to the Drive PLS. The register selected here can be read by the user program or the user interface as a status of the current PLS output.

A-0-0009 Attributes

Data length:	2 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	0
Maximum value:	1024
Default value:	0
Access:	read / write in any Sercos phase

A-0-0020 Maximum Velocity

This parameter sets the maximum programmable velocity for this axis in coordinated, single-axis or velocity mode.

If a coordinated axis is commanded to a value greater than this parameter, the path planner will scale the velocity of this and all coordinated axes to produce a valid coordinated move.

If a single-axis or velocity mode axis is commanded to a value greater than this parameter, error *470 Axis # velocity > maximum* is issued and the axis is stopped based on its parameterized error reaction.

A-0-0020 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005 / min
Minimum value:	+0.001
Maximum value:	1e+07
Default value:	+1000.0
Access:	read / write in any Sercos phase

Note: The drive's bipolar velocity limit parameter (S-0-0091) is the maximum velocity allowed for the motor connected to the drive. Parameter A-0-0020 should always be set to a value less than S-0-0091.

A-0-0021 Maximum Acceleration

This parameter sets the maximum programmable acceleration for an axis in coordinated, single-axis or velocity mode.

Coordinated Accel/Decel

If a coordinated axis is commanded to a value greater than this parameter, the path planner scales the acceleration and deceleration of all coordinated axes to produce a valid coordinated move.

Coordinated Jog Accel

This parameter is also used as the maximum acceleration when jogging a coordinated axis. The maximum decelerarion value is stored in A-0-0022 Maximum Deceleration.

Single Axis and Velocity Jog Accel/Decel

For single-axis or velocity mode, this parameter is only used to set the maximum acceleration and deceleration for jogging.

A-0-0021 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005 / sec²
Minimum value:	+0.01
Maximum value:	1e+07
Default value:	+200.0
Access:	read / write in any Sercos phase

A-0-0022 Maximum Deceleration

This parameter sets the maximum deceleration when jogging a coordinated axis. This parameter is not used for single-axis or velocity mode.

A-0-0022 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005 / sec²
Minimum value:	+0.01
Maximum value:	1e+07
Default value:	+200.0
Access:	read / write in any Sercos phase

A-0-0023 Jog Acceleration

This parameter sets the acceleration and deceleration rate for axis jogging in single-axis or velocity mode. The value entered is a percentage of the maximum acceleration (A-0-0021 Maximum Acceleration).

A-0-0023 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	percentage
Minimum value:	1
Maximum value:	100
Default value:	100
Access:	read / write in any Sercos phase

A-0-0025 Maximum Jog Increment

This parameter sets the maximum distance used for incremental single-axis jogging. Control parameters C-0-0042 World Large Increment and C-0-0043 World Small Increment are percentages that are multiplied with the value in this parameter to produce an incremental jogging distance.

A-0-0025 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005
Minimum value:	+0.0001
Maximum value:	+1000.0
Default value:	+1.0
Access:	read / write in any Sercos phase

A-0-0026 Maximum Jog Velocity

This parameter sets the maximum velocity used for jogging an axis in single-axis mode, velocity mode, or as a joint. Control parameters C-0-0055 Axis Fast Jog Velocity and C-0-0056 Axis Slow Jog Velocity are percentages that are multiplied with the value in this parameter to produce a jogging velocity.

A-0-0026 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005 / min
Minimum value:	+0.0001
Maximum value:	+100000.0
Default value:	+100.0
Access:	read / write in any Sercos phase

A-0-0030 Ratio Mode Master Axis

This parameter defines the axis that will be used as the master for ratio mode. For ELS programs, setting this parameter to 0 assigns ELS Group 1 as the master.

A-0-0030 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	64
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

A-0-0031 Control CAM/Ratio Master Factor (N)

The slave-to-master ratio is defined using two parameters, A-0-0031 and A-0-0032 Control CAM/Ratio Slave Factor (M). This allows the control to normalize the ratio calculation, preserving full system accuracy for repeating-decimal ratios such as 2/3. Both parameters must be set or a run-time error may occur.

In ratio mode, the velocity of the slave is determined by:

$$V_{slave} = V_{master} * (K_{slave} / K_{master})$$

Where:

V_{slave} = Velocity of the slave axis
V_{master} = Velocity of the master axis
K_{slave} = Slave factor set by Parameter A-0-0032
K_{master} = Master factor set by Parameter A-0-0031

A-0-0031 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	--
Minimum value:	-100000.0
Maximum value:	+100000.0
Default value:	+1.0
Access:	read / write in any Sercos phase

A-0-0032 Control CAM/Ratio Slave Factor (M)

Refer to parameter *A-0-0031 Control CAM/Ratio Master Factor (N)* for a description of this parameter.

A-0-0033 Control CAM Stretch Factor (H)

This is the stretch factor (H) for CAM profiles on the control. Every position at the output of the CAM is multiplied by this value.

A-0-0033 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	--
Minimum value:	-100000.0
Maximum value:	+100000.0
Default value:	+1.0
Access:	read / write in any Sercos phase

A-0-0034 Control CAM Currently Active

This is the CAM number that is currently active for this axis. If the CAM is set to 0, the axis directly follows the master axis. CAM activation only takes affect after the master has passed zero degrees or when the master is stopped.

A-0-0034 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	40
Default value:	1
Access:	read / write in any Sercos phase

A-0-0035 Control CAM Position Constant (L)

The value in this parameter is multiplied to the master position according to the following equation.

Scmd = CAM + L * Mcmd

A-0-0035 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	--
Minimum value:	-100000.0
Maximum value:	+100000.0
Default value:	+1.0
Access:	read / write in any Sercos phase

A-0-0036 Ratio Mode Encoder Type

This parameter sets the type of master used for ratio mode. The primary encoder or secondary encoder can be used. The following setting are available:

Setting	Description
0 = Not used or primary feedback is used	selects the master drive's primary feedback, which is the value read from drive parameter S-0-0051
1 = Primary feedback is used	selects the master drive's primary feedback, which is the value read from drive parameter S-0-0051
2 = Secondary feedback is used	selects the secondary feedback, read cyclically from drive parameter S-0-0053 (feedback 2)

Table 15-120: Ratio Mode Encoder Type

A-0-0036 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	2
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

A-0-0037 Ratio Mode Step Rate

This parameter sets the rate used in control ratio mode when the ratio parameters A-0-0031 Control CAM/Ratio Master Factor (N) A-0-0032 Control CAM/Ratio Slave Factor (M) are changed, either directly or through the ***Setup*** ⇒ ***Axis***. If the step rate is set to 0, the ratio will be changed immediately without a ramp.

For example:

The current ratio is 0, the programmed ratio is 10:1, and the step rate parameter is set to 10 units/sec. The ratio will be ramped for one second until it reaches the target value.

A-0-0037 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	units/sec
Minimum value:	+0.0
Maximum value:	1e+07
Default value:	+0.0
Access:	read / write in any Sercos phase

A-0-0038 Ratio Mode Options

In a ratio mode configuration, a master's signal (position or velocity) is evaluated by VisualMotion, taking into account any gear ratio calculations, and a resultant signal is sent to the ratioed slave axis. Any axis in the system can be used as a ratioed axis. ELS Virtual Masters cannot be used in the ratioed axis configuration.

Note: The master number, type and master-to-slave ratio are initially setup under ***Setup* ⇒ *Axis***.. The master-to-slave ratio can be modified in the user program by using the **Ratio** icon. Ratioed slave axes can be switched off a master by writing 0 turns for the slave in the **Ratio** icon.

Bit 1 (Control Mode) of axis parameter A-0-0038 sets up the drive's mode of operation for a ratioed slave axis. Bit 2 (Feedback Mode) determines which master signal will be sent to the control. The values of both bits are evaluated by VisualMotion every Sercos cycle to determine the type of signal (position or velocity command) that will be sent to the ratioed slave axis every Sercos cycle.

```
0000000000000001
               |_ Bit 1

Bit 1:    Control mode  (Default = 0)
Bit 2:    Feedback mode (Default = 0)
Bits 3,4: not used
Bits 5-7: Master Association
```

Fig. 15-19: Bit Description A-0-0038

Bit 1: Control Mode

The Control Mode bit is used to setup the drive's mode of operation (position or velocity) for a ratioed slave axis. The following bit states are defined for bit 1:

0 = Sets the ratioed slave axis in Position Mode (default)

1 = Sets the ratioed slave axis in Velocity Mode

Position Mode (Bit 1=0)

The drive is operating in a closed position loop, with lag or lagless control selected in parameter A-0-0002 Type of Positioning. Master commands sent to the ratioed slave axis by the control will be in position. When the slave axis is switched into ratio mode, it maintains a relative position to the master. A constant Sercos phase difference is maintained between the master and the ratioed slave axis.

Note: Axis parameter A-0-0004 Axis Options bit 11 configures the digital drive's primary or secondary feedback device to close the position loop and provide cyclic feedback from drive parameter S-0-0053.

Velocity Mode (Bit 1=1)

The drive is operating in velocity mode. Master commands sent to the ratioed slave axis by the control will be in velocity. When the slave axis is switched to ratio mode, it follows the velocity of the master. Any position offsets that occur when the slave axis is switched to ratio mode may not be maintained, since no position loop is being closed.

This option is useful for applications where the velocity of the slave needs to be adjusted. The slave velocity may be adjusted in response to a tension loop by changing drive parameter S-0-0037, *Additive Velocity Command Value*.

Bit 2: Feedback Mode

The Feedback Mode bit is used to determine which master signal (position or velocity) is evaluated by the control before it's sent to the ratioed slave axis. The following bit states are defined for bit 2:

0 = The master's position is evaluated by the control (default)

1 = The master's velocity is evaluated by the control

Follow Position (Bit 2=0)

The master's position is evaluated by the control, including any master-to-slave ratio calculations. The resultant is sent to the ratioed slave axis.

If the slave's Control Mode (bit 1) is set to 0 (position mode), the ratioed slave axis follows the master's resultant position.

Note: Bits 1 and 2 of axis parameter A-0-0038 should be set to 0 when relative positioning between the master and the slave is critical.

If the slave's Control Mode (bit 1) is set to 1 (velocity mode), the control calculates a velocity for the ratioed slave axis to follow based on the master's resultant position.

Note: Velocity following is not as accurate as position following due to conversion errors and drive implementation.

Follow Velocity Bit (2=1)

The master's velocity is evaluated by the control, including any master-to-slave ratio calculations. The resultant is sent to the ratioed slave axis.

If the slave's Control Mode (bit 1) is set to 0 (position mode), the control calculates a position for the ratioed slave axis to follow based on the master's resultant velocity.

If the slave's Control Mode (bit 1) is set to 1 (velocity mode), the ratioed slave axis follows the master's resultant velocity.

Note: These options should be selected when problems with position initialization are experienced, or when relative accuracy of position is not required.

Bits 5-7: Master Zero Association

These bits assign the ELS Group number that is used as a ratio mode master when the Master Axis is set to 0. Refer to Ratio Axis Setup for details.

The following table describes the bit assignments:

ELS Group Number	Bit 7	Bit 6	Bit 5
1 (default)	0	0	0
2	0	0	1
3	0	1	0
4	0	1	1
5	1	0	0
6	1	0	1
7	1	1	0
8	1	1	1

Table 15-121: Master Zero Association

A-0-0038 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	00000000 00000011
Default value:	00000000 00000000
Access:	read in any Sercos phase / write in Sercos phase 2

Axis Status (A-0-0100 through A-0-0145)

Parameters A-0-0100 through A-0-0145 provide status values for each configured axis. The values from these parameters are both generated from a running user program or in respond, from the drive, to a program command.

Note: Feedback values are obtained from the drive through the cyclic Sercos telegram rather than through the service channel.

A-0-0100 Target Position

The value in this parameter is the programmed position used by the drive in single-axis mode. Target positions are generated every time a **Move2** icon is encountered in a running user program. VisualMotion writes the programmed target position to this parameter and to drive parameter S-0-0258. Drive parameter S-0-0258 is then used by the drive to move the motor to the target position.

A-0-0100 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005 setting
Minimum value:	-100000.0
Maximum value:	+100000.0
Default value:	generated from running program
Access:	read / write in any Sercos phase

A-0-0101 Commanded Position

The value in this parameter is the commanded position used by the drive in coordinated or ratio mode. Commanded positions are generated in a running user program. VisualMotion updates the commanded position in this parameter and in drive parameter S-0-0047. Drive parameter S-0-0047 is then used by the drive to move the motor to the commanded position.

A-0-0101 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005 setting
Minimum value:	--
Maximum value:	--
Default value:	generated from running program
Access:	read-only

A-0-0102 Feedback Position

This parameter displays the current feedback position value read from the AT telegram every Sercos cycle. Depending on the selected drive options, the AT telegram can contain the value of either S-0-0051, S-0-0053, or S-0-0386.

A-0-0102 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005 setting
Minimum value:	--
Maximum value:	--
Default value:	current feedback value of configured axis
Access:	read-only

A-0-0103 Axis Modulo

This parameter displays the Modulo Value for an Axis. If an axis is configured for Modulo position data, then this parameter will display the value in S-0-0103. If an axis is configured for absolute position data, then this parameter will display 0. This parameter value should be used as an input to function blocks MV_PositionAccumulator and MV_PositionTrigger.

A-0-0103 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

A-0-0110 Programmed Velocity

The value in this parameter is the programmed velocity used by the drive in single-axis mode. Programmed velocities are generated every time a **Velocity** icon is encountered in a running user program. VisualMotion writes the programmed velocity in this parameter and in drive parameter S-0-0259. Drive parameter S-0-0259 is then used by the drive to accelerate the motor at the programmed velocity.

A-0-0110 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005 / min
Minimum value:	+0.0
Maximum value:	1e+07
Default value:	generated from running program
Access:	read / write in any Sercos phase

A-0-0111 Commanded Velocity

The value in this parameter is the commanded velocity used by the drive in velocity mode. Commanded velocities are generated in a running user program. VisualMotion updates the commanded velocity in this parameter and in drive parameter S-0-0036. Drive parameter S-0-0036 is then used by the drive to accelerate the motor at the commanded velocity.

A-0-0111 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005 / min
Minimum value:	--
Maximum value:	--
Default value:	generated from running program
Access:	read-only

A-0-0112 Feedback Velocity

This parameter displays the current feedback velocity value read from the AT telegram every Sercos cycle. Depending on the selected drive options, the AT telegram may contain the value of either S-0-0040.

Note: If S-0-0040 is not configured in the cyclic telegram, the control reads this value from the Sercos service channel.

A-0-0112 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005 / min
Minimum value:	--
Maximum value:	--
Default value:	generated from running program
Access:	read-only

A-0-0120 Programmed Acceleration

The value in this parameter is the programmed acceleration used by the drive in single-axis or velocity mode. Programmed accelerations are generated every time an **Accel** icon is encountered in a running user program. VisualMotion writes the programmed acceleration in this parameter and in drive parameter S-0-0260. Drive parameter S-0-0260 is then used by the drive to accelerate the motor to the programmed acceleration.

A-0-0120 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005 / sec²
Minimum value:	+0.0
Maximum value:	1e+07
Default value:	generated from running program
Access:	read / write in any Sercos phase

A-0-0121 Programmed Deceleration

The value in this parameter is the programmed deceleration used by the drive in single-axis or velocity mode. Programmed decelerations are generated every time an **Decel** icon is encountered in a running user program. VisualMotion writes the programmed deceleration in this parameter and in drive parameter S-0-0359. Drive parameter S-0-0359 is then used by the drive to decelerate the motor.

Note: This parameter is valid for any IndraDrive, EcoDrive Cs using MGP01 firmware and EcoDrive 03 using SGP03 firmware.

A-0-0121 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005 / sec²
Minimum value:	+0.0
Maximum value:	1e+07
Default value:	generated from running program
Access:	read / write in any Sercos phase

A-0-0131 Sercos Control Word

The Sercos control word is automatically configured based on the compiled user program. The Sercos control word is read from parameter S-0-0134 (Master Control Word) and is transmitted cyclically to the control. Refer to S-0-0134 in the relevant *Digital Drive Functional Description* for details.

A-0-0131 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	generated by control
Access:	read-only

A-0-0132 Sercos Status Word

The Sercos status word is read from parameter S-0-0135 (Drive Status Word) and is transmitted cyclically to the control. Refer to S-0-0135 in the relevant *Digital Drive Functional Description* for details.

A-0-0132 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	generated by control
Access:	read-only

A-0-0133 AT Error Count

The AT (drive telegram) error counter is used for troubleshooting of Sercos connections. If the value of this parameter is increasing while being displayed, there may be a noisy Sercos connection or a faulty communication interface on the drive associated with this axis. If two consecutive ATs are invalid, the control issues error 409.

A-0-0133 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0
Access:	read-only

A-0-0140 Mfg. Class 3 Status Word

The function of this parameter will vary based on the connected drive. The following Rexroth digital drives indicate the supported Sercos parameter used for Manufacturer class 3 diagnostics:

- DIAX04 and ECODRIVE03 (supports S-0-0182)
- IndraDrive (supports S-0-0144)

The following table shows when Sercos parameter S-0-0182 or S-0-0144 is automatically placed in the AT:

Operating Mode	Condition
Single-axis	always placed in the AT
Velocity	when A-0-0004, bit 14 = 0
ELS	when A-0-0164, bit 3 = 0
Ratio	when A-0-0164, bit 3 = 0
Control CAM	when A-0-0004, bit 9 = 0 and A-0-0164, bit 3 = 0
Coordinated Articulation	when A-0-0164, bit 3 = 0

Table 15-122: Auto Placement of S-0-0182 or S-0-0144 in to the AT

DIAX04 and ECODRIVE03

For DIAX04 and ECODRIVE, this parameter is the cyclic equivalent of parameter S-0-0182. The control sets the axis status register based on the settings of these bits. Refer to S-0-0182 in the relevant *Digital Drive Functional Description* for details.

IndraDrive

IndraDrive digital drives do not support parameter S-0-0182. For this reason, parameter S-0-0144 will be placed in the AT. However, only four bits will be configured in A-0-0140. The following table shows how the relevant bits in S-0-0144 are configured in A-0-0140:

S-0-0144	A-0-0140	Description
Bit 12	Bit 1	|feedback velocity| < S-0-0124
Bit 13	Bit 6	IZP (position reached)
Bit 14	Bit 8	In_Sych
Bit 15	Bit 11	drive halt and |feedback velocity| < S-0-0124

Table 15-123: S-0-0144 to A-0-0140 Configuration

Bits 0-11 of S-0-0144 are available for configuration by the user.

Note: If S-0-0144 is configured and placed in the AT by the user, bits 12-15 will be automatically configured by the system. Bits 0-11 must be configured by the user.

Refer to S-0-0144 in the relevant *Digital Drive Functional Description* for details.

A-0-0140 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	generated by control
Access:	read-only

A-0-0141 Torque Mode Commanded Torque

This is the cyclic equivalent of drive parameter S-0-0080, Torque Command. This parameter is set from the user program via a parameter transfer. Refer to S-0-0080 in the relevant *Digital Drive Functional Description* for details.

A-0-0141 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	percentage
Minimum value:	--
Maximum value:	--
Default value:	+0.0
Access:	read / write in any Sercos phase

A-0-0142 Torque Feedback (cyclic)

This is the cyclic equivalent of drive parameter S-0-0084, Torque Feedback. This parameter is updated only when an axis is in Torque Mode. Otherwise, the drive's service channel is used. Refer to S-0-0084 in the relevant *Digital Drive Functional Description* for details.

A-0-0142 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	percentage
Minimum value:	--
Maximum value:	--
Default value:	+0.0
Access:	read-only

A-0-0145 Current Motion Type

This parameter displays the current motion type corresponding to the drive's operating mode. This motion type can be the drive's primary or secondary mode of operation. The following motion type are available:

- 0 = disabled
- 1 = single axis
- 2 = coordinated axis
- 3 = velocity mode
- 4 = ratio slave
- 5 = ELS slave
- 6 = torque mode / torque following

A-0-0145 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	0
Maximum value:	6
Default value:	0 (axis selected in program)
Access:	read-only

Electronic Line Shafting (A-0-0150 through A-0-0164)

ELS parameters A-0-0150 through A-0-0164 are written to by an active ELS user program. The various icons that are used are identified in each parameter.

A-0-0150 Programmed Ratio Adjust

This parameter contains the ratio fine adjust used to adjust the velocity or Sercos phase of an ELS configured axis to compensate for mechanical variations between master and slave axes. This value corresponds to drive parameter P-0-0083. It is adjusted from -100 to 300 percent using the **ELSAdj1** icon. If fine adjust is included in the cyclic data, it is updated every Sercos cycle and may be adjusted using a ramp (A-0-0159 Ratio Adjust Step Rate). Otherwise, the drive’s service channel is used.

A-0-0150 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	percentage
Minimum value:	--
Maximum value:	--
Default value:	+0.0
Access:	read / write in any Sercos phase

A-0-0151 Programmed Phase Offset

This parameter contains the programmed Sercos phase offset value used in position synchronization and drive CAM modes. The drive executes a position profile with acceleration (P-0-0142) and velocity limit (P-0-0143) when this parameter is changed. This value corresponds to drive parameter S-0-0048, in the cyclic data, set through the **ELSAdj1** or **CAMAdj** icon.

A-0-0151 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005
Minimum value:	--
Maximum value:	--
Default value:	+0.0
Access:	read / write in any Sercos phase

Note: When dynamic synchronization is enabled in the drive, parameters A-0-0153 and A-0-0155 are not used.

A-0-0153 Control Phase Adjust Average Velocity

This parameter sets the average velocity of the Sercos phase offset move. When a Sercos phase offset is changed, the control performs an absolute positioning move in addition to the ELS master command. This is the target velocity used for the Sercos phase offset adjustment move. The time constant parameter A-0-0155 Control Phase Adjust Time Constant can be used to automatically set the velocity for the Sercos phase offset based on the time of the move.

Note: Peak velocity will be up to twice as large as this value.

A-0-0153 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005 / min or RPM
Minimum value:	+0.0
Maximum value:	1e+06
Default value:	+0.0
Access:	read / write in any Sercos phase

A-0-0155 Control Phase Adjust Time Constant

The control uses a filter to implement a jerk limited profile for the Sercos phase adjust. This parameter sets the amount of time that the move will require, regardless of the position. This can be calculated in the user program so that the Sercos phase adjust is always distributed for the length of one part.

For example:

If parameter A-0-0155 is set to 0, the control uses parameter A-0-0153 Control Phase Adjust Average Velocity, to set the average velocity of the Sercos phase offset move.

Note: Peak velocity will be up to twice as large as this value.

A-0-0155 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	seconds
Minimum value:	+0.0
Maximum value:	+1000.0
Default value:	+0.0
Access:	read / write in any Sercos phase

A-0-0156 Phase Offset Velocity Feedback

This parameter displays the feedback velocity of the Sercos phase offset. The axis must be configured as ELS Sercos phase or ELS CAM slave in the **ELSGrp3** icon.

A-0-0156 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005 / min or RPM
Minimum value:	--
Maximum value:	--
Default value:	+0.0
Access:	read-only

A-0-0157 Current Phase/ Control CAM Master Offset

This parameter displays the current cyclic Sercos phase offset command sent from the control to the drive.

A-0-0157 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005
Minimum value:	--
Maximum value:	--
Default value:	read from control
Access:	read / write in any Sercos phase

A-0-0158 Relative Phase Offset Distance Remaining

This parameter displays the distance remaining on the relative Sercos phase offset. When a continuous Sercos phase offset is in progress, this parameter displays +/- 0.4 * Modulo based on the direction.

A-0-0158 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005
Minimum value:	--
Maximum value:	--
Default value:	+0.0
Access:	read-only

A-0-0159 Ratio Adjust Step Rate

This parameter sets the rate used when the ELS programmed ratio adjust, A-0-0150 Programmed Ratio Adjust, is changed.

If the step rate is set to 0, the ratio adjust will be changed immediately without a ramp.

If ratio adjust is not included in the cyclic data for the drive, the value is not ramped. Refer to *A-0-0004 Axis Options* and *A-0-0180 Optional Command ID #1* for data configuration.

For example:

The current ratio adjust is 0%, the programmed adjust is 10%, and the step rate parameter is set to 10 % / sec. The ratio adjust will be ramped for one second until it reaches the target value.

A-0-0159 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	percentage per second
Minimum value:	+0.0
Maximum value:	+10000.0
Default value:	+0.0
Access:	read / write in any Sercos phase

A-0-0160 Commanded Ratio Adjust

This parameter sets the currently commanded value for ratio adjust, which is updated either gradually or immediately to the programmed ratio adjust value, A-0-0150 Programmed Ratio Adjust.

A-0-0160 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	percentage
Minimum value:	--
Maximum value:	--
Default value:	+0.00
Access:	read / write in any Sercos phase

A-0-0161 Control CAM Programmed Slave Adjust

This parameter sets the target value for the slave Sercos phase adjust (**Sph** in the CAM equation), which is set using the **CAMAdj** icon.

A-0-0161 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005
Minimum value:	--
Maximum value:	--
Default value:	read from active program
Access:	read / write in any Sercos phase

A-0-0162 Control CAM Current Slave Adjust (Sph)

This is the currently commanded value of the slave Sercos phase adjust (**Sph** in CAM equation).

A-0-0162 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005
Minimum value:	--
Maximum value:	--
Default value:	read from active program
Access:	read / write in any Sercos phase

A-0-0163 Control CAM Output Position

This parameter displays the slave position, which is the output of the CAM equation for this axis. It provides a status of the CAM position even when the axis is not synchronized to the CAM. An initial move or Sercos phase offset can be performed with the value read from this parameter before switching into CAM synchronization mode.

A-0-0163 Attributes

Data length:	4 byte data
Data type:	float number
Display format:	signed number with exponent
Units:	A-0-0005
Minimum value:	--
Maximum value:	--
Default value:	read from active program
Access:	read / write in any Sercos phase

A-0-0164 ELS Options

This parameter sets several options for ELS or CAM motion. It's also used to remove certain parameters from the cyclic data. For dynamic synchronization and ramp, set bits 1 and 6.

```
0000000000000001
               |_ Bit 1
Bit 1:  Use drive internal phase offset
Bit 2:  Remove S-0-0048 "Position command value additional"
        from cyclic data
Bit 3:  Remove S-0-0182 "Mfg Class 3 Diag" from cyclic data
Bit 4:  Remove S-0-0258 "Target Position" from cyclic data
Bit 5:  Use service channel for Drive Cam control/status
Bit 6:  Do not automatically set phase offset
Bit 7:  Enable Electronic Pattern Control
Bit 8:  Enable Relative Phase Offset Generator
Bit 9:  Enable Four Drive CAMs (IndraDrive only)
Bit10:  Enable Electronic Motion Profile (IndraDrive only)
```

Fig. 15-20: Bit Description A-0-0164

Bit 1: Use drive internal Sercos phase offset

0 = Phase offset profile is generated by the control

1 = Phase offset profile is generated by the drive (default)

On Rexroth digital drives, the Sercos phase offset for ELS and drive-based CAMs can be generated using parameters P-0-0142, P-0-0143, P-0-0153, and P-0-0155.

Enabling the Sercos phase offset in the drive frees control resources and automatically places parameter S-0-0182 into the cyclic data. Bit 4 (Phase_Adjust) of the relevant axis status register is set to 1 when bit 8 of S-0-0182 is set to 1. Refer to S-0-0182 in the relevant *Digital Drive Functional Description* for details.

Bit 2: Remove S-0-0048 "Position command value additional" (Sercos phase offset) from cyclic data

0 = S-0-0048 is transmitted via the MDT (default)

1 = S-0-0048 is removed from the MDT and transmitted via the service channel

By default, S-0-0048 is placed in the MDT of the cyclic data for ELS slave axes using Sercos phase synchronization. If the Sercos phase offset is never changed while the user program is running, cycle time in the Sercos ring can be conserved by eliminating this parameter from the cyclic data. When the Sercos phase offset is sent through the service channel, the **ELSAdj1** or **CAMAdj** icon will take up to 50ms to execute. If the control is generating the Sercos phase offset profile, the value will change instantly.

Bit 3: Remove S-0-0182 "Mfg Class 3 Diag" from cyclic data

0 = S-0-0182 is transmitted via the AT (default)

1 = S-0-0182 is removed from the AT

Note: Removing S-0-0182 from the cyclic data has an adverse affect on single-axis or velocity mode when using the non-coordinated abort in the **Stop** icon. Since axis status is no longer available, the control can not set the target position equal to the current feedback position upon a non-coordinated abort.

Bit 4: Remove S-0-0258 "Target position" from cyclic data

0 = S-0-0258 is transmitted via the MDT (default)

1 = S-0-0258 is removed from the MDT

Bit 5: Use service channel for Drive CAM control/status

0 = Drive CAM control and status uses drive real time bit (default)

1 = Drive CAM control and status is through service channel

In most applications, this bit should be set to 0. The CAM can then be changed in the following Sercos cycle through the drive real time bits. The drive has two real time bits that are used for CAMs and probe functions. If the CAM does not need to be changed on the fly and more than one probe is needed for registration, this parameter bit can be set.

Bit 6: Do not automatically set Sercos phase offset

0 = Sets Sercos phase offset automatically at synchronization and stores value in A-0-0157 (default)

1 = Phase offset can be initialized with programmed offset (A-0-0151 Programmed Phase Offset) before synchronization.

If this bit is set to 1, the Sercos phase offset can be initialized to any value before the drive is switched into ELS mode. If the bit is set to 0, the control automatically establishes relative Sercos phase synchronization.

Bit 7: Enable Electronic Pattern Control

If this bit is set to 1, the primary operating mode for axes configured as Drive CAMs is enable to Electronic Pattern Control. This is only available for DIAX04 drives using ELS05VRS firmware.

Bit 8: Enable Relative Phase Offset Generator

If this bit is set to a 1, the axis can be commanded a relative ELS phase offsets greater than the axis modulo value. This functionality requires the axis to be configured for a drive-based position synchronization mode and will automatically add P-0-0034 Position Command Additional Actual Value to the SERCOS AT telegram.

Bit 9: Enable Four Drive Cams (IndraDrive Only)

If this bit is set to a 1, the IndraDrive real-time control bits are configured to control the selection of four drive cam profiles. This functionality is only available for the IndraDrive controllers.

Bit 10: Enable Electronic Motion Profile (IndraDrive Only)

If this bit is set to 1, the primary operating mode for the axis is set to Electronic Motion Profile. This operating mode is available for all

IndraDrive functional packages using MPx 04VRS firmware or greater. In addition to the IndraDrive functional packages, SNC and Closed Loop must be selected.

A-0-0164 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	--
Default value:	00000000 00000001
Access:	read in any Sercos phase / write in Sercos phase 2

Axis Feedback Capture (Registration) (A-0-0170 through A-0-0174)

A-0-0170 Probe Configuration Status

This parameter displays the status of the drive feedback capture setup in VisualMotion at program activation or execution. The control uses the Sercos probe functions and real-time bits along with the event system to allow user programs to perform registration functions. Rexroth digital drives provide two probe inputs that can be used for capturing the feedback position.

```
0000000000000001
               |_ Bit 1

Bit 1: Probe 1 positive edge enabled
Bit 2: Probe 1 negative edge enabled
Bit 3: Probe 2 positive edge enabled
Bit 4: Probe 2 negative edge enabled
```

Fig. 15-21: Bit Description A-0-0170

A-0-0170 Attributes

Data length:	2 byte data
Data type:	binary number
Display format:	binary
Units:	--
Minimum value:	00000000 00000000
Maximum value:	--
Default value:	00000000 00000001
Access:	read-only

A-0-0171 Probe 1 Positive Captured Value

This parameter displays the last captured value for the probe 1 positive edge (0 ⇒1) input on the drive. Upon either a positive or a negative transition of a probe input, the drive captures the position into the cyclic data. Since the captured feedback positions must be included in the Sercos cyclic data telegram, the probe setup icon must be included in the user program for each drive that will use the probe function.

A-0-0171 Attributes

Data length:	4 byte data
Data type:	integer
Display format:	signed decimal
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0.0000
Access:	read-only

A-0-0172 Probe 1 Negative Captured Value

Refer to *A-0-0171 Probe 1 Positive Captured Value* for a description of this parameter.

A-0-0173 Probe 2 Positive Captured Value

Refer to *A-0-0171 Probe 1 Positive Captured Value* for a description of this parameter.

A-0-0174 Probe 2 Negative Captured Value

Refer to *A-0-0171 Probe 1 Positive Captured Value* for a description of this parameter.

Optional Sercos Data (A-0-0180 through A-0-0196)

A-0-0180 Optional Command ID #1

VisualMotion automatically configures the MDT (S-0-0024, Configuration List of the Master Data Telegram) based on the mode of operation specified in the user program and the axis parameter settings during system initialization (Sercos phase 2 to Sercos phase 3).

In addition to the configured MDT, this parameter allows the user to add an additional drive parameter from S-0-0188 to the MDT by entering the parameter IDN number.

For example:

S-0-0040 is entered as **40**

P-0-0053 is entered as 53 + 32768 or **32821**

Note: Since the MDT is configured based on the mode of operation and axis parameters, there might not be enough room in the telegram for an additional drive parameter. If this is the case, the control will issue an error.

Parameter A-0-0180 works in conjunction with parameter A-0-0190 Command Data #1. A-0-0180 identifies which drive parameter is being added to the MDT and A-0-0190 stores the value for that parameter.

For example:

To update the torque limit in real-time, set **A-0-0180** to 92 (Torque Limit). While the drive is in Phase 4, the value in Parameter **A-0-0190** is sent cyclically to the drive, and can be written using the **Param1** icon in the user program.

A-0-0180 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

A-0-0181 Optional Command ID #2

Refer to *A-0-0180 Optional Command ID #1* for a description of this parameter.

A-0-0182 Optional Command ID #3

Refer to *A-0-0180 Optional Command ID #1* for a description of this parameter.

A-0-0183 Optional Command ID #4

Refer to *A-0-0180 Optional Command ID #1* for a description of this parameter.

A-0-0184 Optional Command ID #5

Refer to *A-0-0180 Optional Command ID #1* for a description of this parameter.

A-0-0185 Optional Feedback ID #1

VisualMotion automatically configures the AT (S-0-0016, Custom Amplifier Telegram Configuration List) based on the mode of operation specified in the user program and the axis parameter settings during system initialization (Sercos phase 2 to Sercos phase 3).

In addition to the configured AT, this parameter allows the user to add an additional drive parameter from S-0-0187 to the AT by entering the parameter IDN number.

For example:

S-0-0040 is entered as **40**

P-0-0052 is entered as 52 + 32768 or **32820**

Note: Since the AT is configured based on the mode of operation and axis parameters, there might not be enough room in the telegram for an additional drive parameter. If this is the case, the control will issue an error.

Parameter A-0-0185 works in conjunction with parameter A-0-0195 Feedback Data #1. A-0-0185 identifies which drive parameter is being added to the AT and A-0-0195 displays the current value for that parameter.

For example:

By default, feedback velocity is received through the service channel. To obtain the feedback velocity in real-time, set A-0-0185 to 40 (Feedback Velocity). While the drive is in Phase 4, the value in Parameter A-0-0195 is updated, and can be read using the **Param1** icon in the user program.

A-0-0185 Attributes

Data length:	2 byte data
Data type:	unsigned integer
Display format:	unsigned decimal
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0
Access:	read in any Sercos phase / write in Sercos phase 2

A-0-0186 Optional Feedback ID #2

Refer to *A-0-0185 Optional Feedback ID #1* for a description of this parameter.

A-0-0187 Optional Feedback ID #3

Refer to *A-0-0185 Optional Feedback ID #1* for a description of this parameter.

A-0-0188 Optional Feedback ID #4

Refer to *A-0-0185 Optional Feedback ID #1* for a description of this parameter.

A-0-0189 Optional Feedback ID #5

Refer to *A-0-0185 Optional Feedback ID #1* for a description of this parameter.

A-0-0190 Command Data #1

This parameter displays the real-time value that corresponds to the parameter identified in A-0-0180 Optional Command ID #1. Changes to this parameter will affect the value of the drive parameter set in A-0-0180.

Note: When using the **Param1** icon to transfer a value for the parameter stored in A-0-0180, be aware of the attribute for that parameter. If the parameter identified in A-0-0180 is of type float, transfer a float value. For any other types (integer, hexadecimal, binary), transfer an integer. A simple way of identifying this type is to view the drive parameter via the Parameter Overview Tool.

A-0-0190 Attributes

Data length:	4 byte data
Data type:	attribute of parameter in A-0-0180
Display format:	attribute of parameter in A-0-0180
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0
Access:	read / write in any Sercos phase

A-0-0191 Command Data #2

Refer to A-0-0190 Command Data #1 for a description of this parameter.

A-0-0192 Command Data #3

Refer to A-0-0190 Command Data #1 for a description of this parameter.

A-0-0193 Command Data #4

Refer to A-0-0190 Command Data #1 for a description of this parameter.

A-0-0194 Command Data #5

Refer to A-0-0190 Command Data #1 for a description of this parameter.

A-0-0195 Feedback Data #1

This parameter displays the real-time value that corresponds to the parameter identified in *A-0-0185 Optional Feedback ID #1*.

A-0-0195 Attributes

Data length:	4 byte data
Data type:	attribute of parameter in A-0-0185
Display format:	attribute of parameter in A-0-0185
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	0.0000
Access:	read-only

A-0-0196 Feedback Data #2

Refer to A-0-0195 Feedback Data #1 for a description of this parameter.

A-0-0197 Feedback Data #3

Refer to A-0-0195 Feedback Data #1 for a description of this parameter.

A-0-0198 Feedback Data #4

Refer to A-0-0195 Feedback Data #1 for a description of this parameter.

A-0-0199 Feedback Data #5

Refer to A-0-0195 Feedback Data #1 for a description of this parameter.

Multiplexing Parameters (A-0-0200 through A-0-0203) (DKC 2.3 only)

VisualMotion configures the MDT and AT based on the modes of operation specified in the application program and Axis parameter settings during system initialization (Sercos phase 2 to Sercos phase 3).

In DKC 2.3 drives, if the maximum telegram length is exceeded the system will automatically enable the multiplex (mux) channel. The system will then populate the mux with up to 5 IDNs base on the system and user selections. The system will also populate the cyclic portion of the telegram with up to 3 IDNs (8 bytes) that must be cyclically transferred. This is based on the system and user selections.

A-0-0200 MDT Multiplex Selection List (DKC 2.3 only)

This parameter contains a list of all supported MDT data identification numbers (IDN) usable within the VisualMotion system. It is automatically transferred to each DKC2.3 drive (S-0-0370) when mux support is required. This is a read only list and will be uploaded to display valid selections for programming the Optional Command Data channel (parameters A-0-0180…182).

Note: Multiplex parameters are only support with DKC2.3 digital drives.

The contents of this parameter can be viewed by double clicking on the parameter number in the *Parameter Overview* tool.

A-0-0200 Attributes

Data length:	variable length 4 byte data
Data type:	extended character set
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	32 list elements
Default value:	--
Access:	read-only

A-0-0201 AT Multiplex Selection List (DKC 2.3 only)

This parameter contains a list of all supported AT data identification numbers (IDN) usable within the VisualMotion system. It is automatically transferred to each DKC2.3 drive (S-0-0371) when mux support is required. This is a read only list and will be uploaded to display valid selections for programming the Optional Command Data channel (parameters A-0-0190…191).

Note: Multiplex parameters only support with DKC2.3 digital drives.

The contents of this parameter can be viewed by double clicking on the parameter number in the *Parameter Overview* tool.

A-0-0201 Attributes

Data length:	variable length 1 byte data
Data type:	string array
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	32 list elements
Default value:	--
Access:	read-only

A-0-0202 MDT Multiplex Ident List (DKC 2.3 only)

This parameter list contains the Idents that the system has automatically placed in the MDT mux circular queue.

A-0-0202 Attributes

Data length:	variable length 1 byte data
Data type:	string array
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	10 list elements
Default value:	--
Access:	read-only

A-0-0203 AT Multiplex Ident List (DKC 2.3 only)

This parameter list contains the Idents that the system has automatically placed in the AT mux circular queue.

A-0-0203 Attributes

Data length:	variable length 1 byte data
Data type:	string array
Display format:	text
Units:	--
Minimum value:	--
Maximum value:	10 list elements
Default value:	--
Access:	read-only

Axis Parameter Lists (A-0-2000 and A-0-2001)

A-0-2000 List of All Parameters

This parameter contains a list of all axis parameters that are part of the current firmware version. The contents of this parameter can be viewed by double clicking on the parameter number in the *Parameter Overview* tool.

A-0-2000 Attributes

Data length:	variable length 4 byte data
Data type:	unsigned integer
Display format:	IDN (parameter number)
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

A-0-2001 List of Required Parameters

This parameter contains a list of all required axis parameters that are part of the archive / restore function. The contents of this parameter can be viewed by double clicking on the parameter number in the *Parameter Overview* tool.

A-0-2001 Attributes

Data length:	variable length 4 byte data
Data type:	unsigned integer
Display format:	IDN (parameter number)
Units:	--
Minimum value:	--
Maximum value:	--
Default value:	--
Access:	read-only

16 Registers

16.1 Overview

Rexroth VisualMotion uses registers as a method for communication and system monitoring between the control (using I/O) and an external system. Rexroth VisualMotion provides 1024 registers. Some registers are reserved for control and system functions and others are recommended as defaults. The remaining registers are available for use by the programmer. Registers can be viewed by selecting ***Data ⇒ Registers*** from Rexroth VisualMotion Toolkit's main menu. A breakdown of Rexroth VisualMotion registers can be found in Table 16-1.

Register	Register Label	Availability
1	System Control	Reserved for System
2-5	Task A-D Control	Reserved for System
6	System Diagnostic Code	Reserved for System
7-10	Task A-D Jog Control	Reserved for System
11-18	Axis Control 1-8	Reserved for System
19	Fieldbus_PLC Status	Reserved for System
20	Fieldbus_PLC Diagnose	Reserved for System
21	System Status	Reserved for System
22-25	Task A-D Status	Reserved for System
27	Initialization Task Control	Reserved for System
28	Initialization Task Status	Reserved for System
31-38	Axis Status 1-8	Reserved for System
40	Link Ring Status	Reserved for System
41-42	Link Ring Data	Reserved for System
44	PPC_Onboard_XI	Reserved for System
45	PPC_Onboard_XO	Reserved for System
50	Ethernet 1 Status	Reserved for System
51	Standard Message Count	Reserved for System
52	Cyphered Message Count	Reserved for System
53	Invalid Count	Reserved for System
54	SIS Message Count	Reserved for System
55	Ethernet 2 Status	Reserved for System
86	PMG Control	Reserved for System
87	PMG Status	Reserved for System
88 and 89	Task A Extend Event Control	Reserved for System
90 and 91	Latch and Unlatch	Reserved for System
92-94	Mask BTC06 Key Functionality	Reserved for System
95-97	BTC06 Teach Pendant Status	Reserved for System
98 and 99	BTC06 Teach Pendant Control; Task A-B, C-D	Reserved for System
100	IO_Box_Inputs	System Default
120	IO_Box_Outputs	System Default

Register	Register Label	Availability
140	ELS Master Control	Reserved for System
141	ELS Master Status	Reserved for System
150 and 151	Virtual Master 1 & 2 Control	System Default
152-159	ELS Groups 1 – 8 Control	System Default
197	Coordinated Articulation Synchronized Mode Control	System Default
198	Coordinated Articulation Local Mode Control	System Default
209-240	Axis Control 9-40	Reserved for System
241 and 242	Virtual Master 1 & 2 Status	System Default
243-250	ELS Groups 1 – 8 Status	System Default
288	Coordinated Articulation Synchronized Mode Status	System Default
289	Coordinated Articulation Local Mode	System Default
309-340	Axis Status 9-40	Reserved for System
441-464	Axis Control 41-64	Reserved for System
541-564	Axis Status 41-64	Reserved for System

Table 16-1: Control Registers

Register 1: System Control

System Control register bits are dedicated to system supervisory control functions.

Register	Register Label Name	Bit	Function
1	System_Control	1	Parameter Mode
		2	Not Used
		3	nEmergency Stop
		4	Not Used
		5	Clear All Errors
		6	Pendant Live-Man
		7	Rebuild Double Ring
		8	Activate Program
		9	Program Select LSB
		10	Program Select Bit_2
		11	Program Select Bit_3
		12	Program Select MSB
		13	Not Used
		14	Pendant Enable
		15	Pendant Level LSB
		16	Pendant Level MSB

Table 16-2: Register 1: System Control

Bit 1: Parameter Mode/ nRun Mode

A low-to-high (0-1) transition of this bit switches the system into Parameter Mode. All user tasks are immediately interrupted. The system is switched into parameter mode, and the drives are switched into Sercos phase 2.

Note: This transition is possible if this bit is not mapped to a different register bit by the PLC. In addition, when the system is forced into Sercos phase 2 (parameter mode) or Sercos phase 4, a high-to-low transition of this bit will not switched the control from phase 2 to phase 4. Refer to C-0-0011 for details.

A high-to-low transition (1-0) re-initializes the system and switches it to Run Mode. Parameter initializations are performed and the drives are switched from phase 2 to phase 4. If there are no errors, the user tasks are ready to operate.

Bit 3: nEmergency Stop

This input is active low (0 = Emergency Stop). When set to (0), all user tasks are stopped except those selected to run during errors. All motion is stopped, and the drives are set to zero velocity and disabled. When set to (1), the emergency stop condition has been corrected, and the tasks can run if there are no other errors.

Bit 5: Clear All Errors

A low-to-high transition clears any existing system, task and drive errors as long as the condition that cause the error is resolved.

Bit 6: Pendant Live Man

This bit should be mapped to the teach pendant live man switch when a teach pendant is used. When set to (0), no motion can be initiated from the teach pendant and any motion in progress is immediately stopped. When set to (1), (*live-man closed)* the teach pendant can jog, start, and stop motion.

Bit 7: Rebuild Double Ring

This bit is used to reinitialize primary and secondary Link Rings when redundancy has been lost. This bit is set high (1) at each node in the Link Ring to simultaneously clear the Redundancy_lost bits after any Sercos cable problems have been corrected. This bit is functional only in Sercos Phase 4.

Bit 8-12: Activate Program and Binary Program Select

The active program can optionally be changed using I/O bits 9-12 in the System Control register. A transition from (0) to (1) on Activate Program bit 8 will start the program based on bits 9-12. These bits correspond to the program number (from 1 to 10). If the Activate Program bit is high at power-up, the program selected with the Binary Program Select bits will be activated.

Note: User interface (Rexroth VisualMotion, Teach Pendant, etc.) selections take precedence over these bits. The actual active program is acknowledged in System Status Register bits 9-12

Example

To activate program 5, set the bits as follows:

Bit 9: 1

Bit 10: 0

Bit 11: 1

Bit 12: 0

Then Bit 8 requires a transition from 0 to 1.

Bit 14: Pendant Enable

This bit toggles control of tasks and jogging between the teach pendant and the I/O system. When set to (0), the system I/O and control registers are in control. When set to (1), the teach pendant assumes control of system functions, forcing all relevant bits in the control registers.

Bits 15-16: Pendant Access Level

These bits provide access protection for teach pendant menus. The protection levels are defined per-menu to provide restricted access to data. If a menu's protection level exceeds the value of these bits, the menu can be viewed but not edited.

Registers 2-5: Task Control

Task Control registers 2, 3, 4 and 5 are dedicated to task control for Tasks A, B, C and D respectively.

Register	Register Label Name	Bit	Function
2	Task A Control	1	Mode:Auto /nManual
3	Task B Control	2	Override Automatic Start
4	Task C Control	3	Not Used
5	Task D Control	4	Single Step
		5	Not Used
		6	Cycle Start /Resume
		7	nTask Stop
		8	Not Used
		9	Task Event Trigger
		10	Trace Enable
		11	Breakpoint Enable
		12	Sequencer Single Step
		13	Step Sequence Function
		14	Not Used
		15	Coordinate Fast Stop
		16	Not Used

Table 16-3: Registers 2-5: Task Control

Bit 1: Mode: Automatic/ nManual

This bit selects the mode of operation for a task. When set to (0), the task is in Manual Mode, the user program does not run, and manual jogging is enabled. When set to (1), the task is in Automatic Mode and is ready to begin execution.

Coordinated, Single Axis and Velocity Modes

Bit 1 = 1
Switch to Automatic

The instruction pointer is reset to the beginning (Start Icon) of the program, and all events become inactive. At the next (0-1) transition of the cycle start bit when the cycle stop bit is (1), the program starts running.

Bit 1 = 0
Switch to Manual

The user task and any events associated with it stop execution immediately. Any motion associated with the task is immediately decelerated to zero velocity. Coordinated, single-axis, and velocity axes are stopped using the maximum deceleration.

ELS Mode

Bit 1 = 1
Switch to Automatic

Rexroth VisualMotion's program flow resets at the beginning (Start Icon) of the program. Virtual Masters 1 and 2 are stopped at the programmed deceleration. All ELS Groups with active real or group input masters are switched to local mode.

Bit 1 = 0
Switch to Manual

The ELS master is stopped using the E-Stop deceleration. The instruction pointer is set to where the program was stopped, but is reset to the beginning of the program when the task is returned to Auto Mode.

Bit 2: Override Automatic Start

Each task in a project can be individually configured to start immediately upon exiting parameter mode or after clearing an error when task parameter T-0-0002, bit 4 (Automatically Start Task) is set to 1.

When this bit is set to (1) and T-0-0002, bit 4 = 1, the "Automatically Start Task" option is overridden and the task is disabled. The other task control register bits, such as Cycle Start/Resume, Single Step Select, etc., are enabled.

When this bit is set to (0) and T-0-0002, bit 4 = 1, the "Automatically Start Task" option is enabled and the task control register bits are disabled.

Note: For running tasks, the setting of the Override Automatic Start bit requires a state change from one of the other task control register bits in order to enable or disable the function.

Bit 4: Single Step Select

When set to (1), the task is placed in Single Step mode. Each positive (0 to 1) transition of the Cycle Start bit executes one user task program instruction then pauses (providing that the system is in Automatic mode and nTask Stop is inactive). If this bit is set while a task is running, the current instruction completes. The task then pauses and waits for a Cycle Start transition.

Event functions cannot be single-stepped. If one or more events have been started or queued by executing the current instruction, the events will always run to completion before pausing the user task program.

When set to (0), Single Step mode is disabled. When Single Step is zero, normal cycles begin at the next positive transition of the Cycle Start bit (providing that automatic mode is true and nTask Stop is false).

Note: If the task auto start option is set (T-0-0002, bit4=1), single step functionality will not work. Either disable the task auto start option or set this control register's bit 2 (Override Automatic Start) to 1 to enable the single step functionality.

Bit 6: Cycle Start/Resume

A low-to-high transition (0-1) start executing the user task starts executing at the current instruction, if the task is in automatic mode, the nTask Stop bit is (1), and there are no errors. It is also used to resume the task after a task stop and to restart the task after entering automatic mode. If single-stepping is enabled, the next instruction is executed with each positive transition. A low-to-high (0-1) transition is required to start or resume the task

Bit 7: nTask Stop

When Bit 7 = 0, the **n**Task is stopped. A high-to-low transition (1-0) stops the task program at the end of the current instruction. All types of motion are halted and can be resumed at the next Cycle Start. The **n**Task Stop bits function as a “Pause”. When the bit is set to 0, the control pauses execution of the task and the instruction pointer remains at the current instruction. When the bit is again = 1 and the cycle start bit is toggled, the task resume execution at the current instruction.

All types of events will execute during a cycle stop state. Only the main program flow in tasks A, B, C and D is affected.

In **Coordinated motion**, the nTask Stop bit will decelerate the currently active segment point at the point's programmed deceleration percentage.

CAUTION

Coordinated motion segment points are programmed with individual acceleration and deceleration percentage rates. Using the nTask Stop bit might not decelerate the axis as quickly as desired. If an immediate stop condition is required, use the Coordinate Fast Stop function (bit 15).

Motion is then paused on the current segment. All distance and time-based events remain active. As long as the task is in automatic mode, the next 0 to 1 transition on the cycle start (bit 6) will resume motion and complete all pending segments.

Single axis motion halts each axis in the task by decelerating the axes to zero velocity while retaining target position. The previous state of the GO command is saved until the next cycle start. If the GO command was active, motion will be resumed at the next cycle start in automatic mode. All normal and repeating events remain active. All **Velocity** mode axes are decelerated to zero velocity if ramping is enabled, or set to zero velocity if step command is selected. All events remain active.

The **ELS** master axis is decelerated to zero velocity. The previous state of the GO command is saved until the next cycle start. If the GO command was active, the master will be commanded the last programmed velocity. All events on the ELS slaves remain active. The slave axes remain synchronized to the master if synchronization was enabled. Because the control has no control over a real master, motion of the slaves of a real master cannot be changed. The operation of **Torque** mode axes during a **n**Task Stop is not defined at this time.

Note: *nTask Stop does nothing to assure that the system is in a safe or known condition to stop. nTask Stop simply completes the current instruction, then ramps down motion in the task.*

Bit 9: Task Event Trigger

This bit is reserved as an Event Interrupt Input for each task. Each low-to-high (0 to 1) transition of this input will trigger an event to the corresponding task. This event type can be used to start a process or to respond to an external event.

In the event table, Type 6 selects an Interrupt Input event. The event/trigger (arm event) instruction enables the interrupt input. The event/done (disarm event) instruction is used to disable the input. The control scans the input every 4ms and starts an event upon a low-to-high transition. The event function will take priority over the user tasks, allowing quick response to an external input. IndraLogic can be used to reverse the logic of the interrupt input, or to direct other external inputs to it. Logic in the event function can then scan the multiple inputs to determine the source of the interrupt.

Bit 11: Breakpoint Enable

When this bit is set to (1), the breakpoint enabled in task parameter T-0-0137 is active. When program flow reaches the breakpoint, the task is stopped. When this bit is set to (0), the program executes normally, without breakpoints.

Bit 12: Sequencer Single Step

This bit places the control into Sequence Single Step mode. As long as this bit is (1), a low-to-high (0-1) transition on the cycle start bit causes the program to be stopped after each sequencer step is executed. If this bit is (0), the sequencer executes normally.

Bit 13: Step Sequence Function

This bit places the control into Function Single Step mode. As long as this bit is (1), a low-to-high (0-1) transition on the cycle start bit causes the program to be stopped after each sequencer function is executed. If this bit is (0), the sequencer executes normally.

Bit 15: Coordinate Fast Stop

This bit is used to perform a controlled stop of all task motion using the task's speed (T-0-0020), acceleration (T-0-0021) and deceleration (T-0-0022) parameters. A low-to-high transition (0-1) actives this function, halting all task motion while maintaining positioning. A high-to-low transition (1-0) deactivates this function, continuing task motion from the stopped position.

Cycle Control Considerations

Cycle Stop in User Program

A cycle stop implies that a task's motion cycle has completed and the system is at a safe place to halt. The nTask Stop bit cannot always be used for this purpose, since it only stops task instruction execution and commands the drives to ramp down.

A nTask Stop signal may lose track of user task activity that is related to axis or segment position or time. In addition, the control's path planner may have several queued segments or events. Queued events always continue execution until completion. This may result in the system position and I/O losing synchronization with your programmed sequence of task instructions. If you attempt to simply restart motion, the results may not be predictable.

If each of your tasks require a cycle stop capability, a separate user task I/O bit should be user configured into the I/O system for each task needing a cycle stop. Your program then tests the associated I/O bit from within your task program. Use the condition of the I/O bit to branch to a program routine that halts motion and establishes a known system state. Since you are programming a unique system, only you can determine a safe system condition.

System Parameter Mode

A switch to Parameter Mode immediately disables all user tasks, and switches the control and axes to Sercos Phase 2.

System Shutdown Errors

A shutdown error disables all drives, stops coordinated and single-axis motion, and puts the task into manual mode. All control bits are left at their current state.

Programmed End of Task

A task program that reaches and executes the task/end instruction or the Finish Icon has the same effect as activating the *n*Task Stop I/O line.

Register 6: System Diagnostic Code

This status register shows the current control diagnostic code in Motorola 16-bit format. This register is displayed as a 3 digit code when ***Decimal*** is the selected ***Format*** under ***Data ⇒ Registers***.

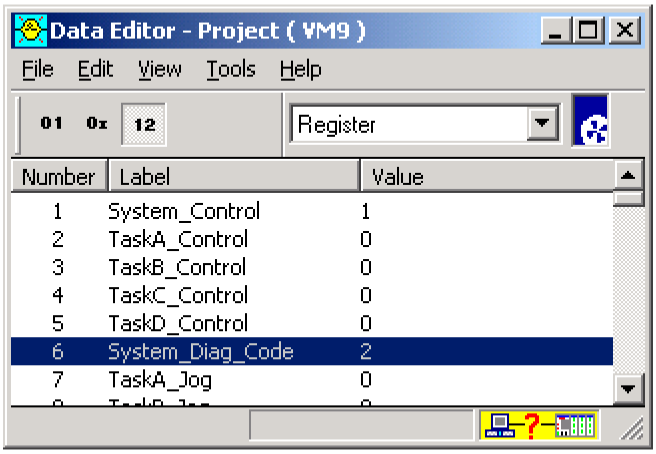


Fig. 16-1: Displaying Register 6

Registers 7-10: Task Jog Control

Registers 7, 8, 9 and 10 are dedicated for jogging control of Tasks A, B, C and D respectively.

Single-Axis and Velocity Mode Jogging

Any axis in single-axis or velocity mode can be jogged independently when its associated task is in manual mode. Single-axis and velocity mode axes are jogged using the Axis Control Registers. Motion is started with a low-to-high transition on either the jog forward or the jog reverse bit. Motion is stopped when both jog bits are low, when both jog bits are high, when the task mode selection changes, or when a travel limit or incremental distance has been reached. Jog mode (continuous or incremental) and jog speed or distance is selected in the Task Jog Register for the task associated with the axis.

Coordinated Jogging

Coordinated axes may be jogged in any direction while a task is in manual mode using these Task Jog Registers. The coordinate to jog, type of jog, and parameters to use are selected using bits in this register.

Before jogging can begin, the coordinated axis must be selected with bits 9, 10, or 11 of the relevant task control register. Jogging begins when either bit 2 (Coordinated Jog Forward) or bit 3 (Coordinated Jog Reverse) is transitioned from 0 to 1. The axis will move the distance calculated by C-0-0042 and T-0-0025 with every 0-1 transition of the Coordinated Jog Forward or Reverse bits.

Note: When a drive is referenced, the G*P 11 control firmware detects the travel limits for Kinematic #1 only. Once started, jogging automatically stops before the drive travel limit is reached. For all other Kinematics, if the drive is referenced, jogging will continue if commanded until the drive issues a travel limit fault.

Register	Register Label Name	Bit	Function
7 8 9 10	Task A Jog Task B Jog Task C Jog Task D Jog	1	Continuous nStep
		2	Coordinate Jog Forward
		3	Coordinate Jog Reverse
		4	Jog Type LSB
		5	Jog Type MSB
		6	Distance Speed
		7 and 8	Not Used
		9	Jog X Coordinate
		10	Jog Y Coordinate
		11	Jog Z Coordinate
		12 - 16	Not Used

Table 16-4: Registers 7-10: Task Jog

Bit 1: Mode: Continuous/nStep

This bit is used to select the jog mode for both coordinated and single-axis jogging. The mode takes effect when the next jog is started with a transition on the jog forward or jog reverse bit.

0 = nStep jogging. Motion stops after the large or small distance is reached, or when the jog bit is set to 0.

1 = continuous jogging. Motion stops when the travel limit is reached on an axis or when the jog bit is set to 0.

Bit 2 and 3: Coordinated Jog Forward and Reverse

Coordinated Motion

A low-to-high (0-1) transition on this bit while a task is in manual mode causes motion to start in the positive (Bit 2) or negative (Bit 3) direction for the coordinate selected in bits 9 to 14. A high-to-low (1-0) transition immediately stops the motion.

Bits 4 and 5: Jog Type

Bits 4 and 5 select the type of coordinated jogging for the next jog motion.
World Jog: jogs the axes in the world coordinate selected by bits 9, 10, and 11.

Bit 5	Bit 4	Jog Type Selected
0	0	World Jog
0	1	Joint Jog
1	1	not used

Table 16-5: Task Jog Type

Note: Selecting an invalid jog type will issue a warning message.

Bit 6: Distance/Speed (Large/nSmall, Fast/nSlow)

When bit 6 is set to 0 before a continuous jog, the slow jog speed is selected. For an incremental jog, the small distance and slow speed are selected.

When bit 6 is set to 1 before a continuous jog, the fast jog speed is selected. For an incremental jog, the large distance and fast speed are selected.

Bits 9-11: Jog Coordinate and Joint

These bits select the coordinate or joint that will be jogged when the jog forward or jog reverse bits are activated. Only one of these bits should be set, since jogging is allowed in only one coordinate at a time.

Bit number	World and Tool Jog	Joint Jog
9	X Coordinate	Joint 1
10	Y Coordinate	Joint 2
11	Z Coordinate	Joint 3

Table 16-6: Jog Coordinate and Joint

When jogging in world coordinates, motion will be generated parallel to the selected X, Y, or Z coordinate according to bits 9 through 11. For example, setting bit 9 high and bits 10 and 11 low enables jogging parallel to the X-axis.

For jogging of individual joints, the axis to jog is selected in bits 9 through 14. For example, if axes 2, 3, and 4 are used in coordinated motion, bit 9 selects axis 2, and bit 11 selects axis 4. The jog speed used is a percent of maximum axis velocity.

If an invalid jog type is selected, a warning message is issued.

Registers 11-18, 209-240, 441-464: Axis Control

Registers 11-18 are used for axes 1-8. Registers 209-240 are used for axes 9-40. Registers 441-464 are used for axes 41-64. These Registers are reserved for axis control. These axes correspond to the Sercos drive address on the fiber-optic communication loop. Any drive-controlled axis (single-axis, velocity) may be jogged independently when its associated task is in manual mode.

Jog mode (continuous or incremental) and jog speed or distance must be selected in the Task Jog Register for the task associated with the axis. Motion starts with a low-to-high transition on either the jog forward or the jog reverse bit. Motion stops when both jog bits are low, when both jog bits are high, when the task mode selection changes (e.g., from manual to automatic mode), or when a travel limit or incremental distance has been reached.

Register	Register Label Name	Bit	Function
11-18	Axis 1-8	1	Disable Axis
209-240	Axis 9-40	2	Jog Forward
441-464	Axis 41-64	3	Jog Reverse
		4	Synchronized Jog
		5-16	not used

Table 16-7: Registers 11-18: Axis Control

Bit 1: Disable Axis

When set to (1), all axis motion is disabled based on the set *Drive Disable Method* of axis parameter A-0-0004, bit 12:

- When A-0-0004, bit 12 is set to (0), the drive stops the motor according to the value in P-0-0119. By default P-0-0119 = 0; the drive immediately commands the motor to zero velocity using full torque (limited by S-0-0092) before disabling torque and applying the brake (if equipped).
- When A-0-0004, bit 12 is set to (1), the drive immediately disables torque, causing the motor to coast to a stop using its own inertia.

When set to (0), motion is enabled, provided other conditions allow it and there are no errors on the drive or the control. The axis resumes any motion commands from the control.

Bit 2 and 3: Jog Forward (bit 2) and Reverse (bit 3)

A low-to-high (0-1) transition on this bit while an axis is in manual mode causes motion to start in the positive (bit 2) or negative (bit 3) direction. A high-to-low (1-0) transition issues a drive halt (AH) command and stops motion based on the drive's predefined deceleration rate. Refer to *Drive Halt Setup (AH)* in Chapter 11, *Error Reaction*, of volume 1 for details.

Motion is stopped when both jog bits are low, when both jog bits are high, when the task mode selection changes, or when a travel limit or incremental distance has been reached. Jog mode (continuous or incremental) and jog speed or distance is selected in the Task Jog Register for the task associated with the axis.

Bit 4: Synchronized Jog

When set to (1) the following cases are enabled:

- For Ratio Mode, the axis will remain synchronized to its "master".
- For Torque Mode / Torque Following Mode, the axis will be switched to Velocity mode.

Register 19: Fieldbus_PLC Status

Register 19 holds the information for "Fieldbus Status." The register information can be referenced in a Rexroth VisualMotion application program to respond to the status of each bit. The use of these bits is application-dependent. Table 16-8 below contains the bit assignment for the diagnostic object 5ff2.

Register	Register Label Name	Bit	Function
19	Fieldbus/PLC Status	1	FB Init. OK, LSB
		2	FB Init. OK, MSB
		3	Not Used
		4	FB Slave Ready
		5	Non Cyclic Ready
		6	Not Used
		7 -12	Not Used
		13	Not Used
		14	Register Data Valid
		15	Cyclic Data Valid
		16	Not Used

Table 16-8: Register 19: Fieldbus/PLC Status

Bit 1 and 2: Fieldbus Initialization OK; LSB and MSB

Status bits for the internal DPR (Dual-Port RAM) communication between the Fieldbus slave and the PPC-R.

Bit 1: FB Init. OK, LSB (least significant bit)

Bit 2: FB Init. OK, MSB (most significant bit)

Bit 2 (Control)	Bit 1 (Fieldbus)	Description
0	0	A reset has been executed on the DPR, **or** neither the PPC-R nor the fieldbus card has initialized the DPR.
0	1	The DPR is initialized by the fieldbus card, but **not** yet by the PPC-R.
1	0	The DPR initialization is complete. DPR has been initialized by the fieldbus card and PPC-R. Fieldbus to PPC-R communications system is ready.
1	1	Fieldbus to PPC-R communications system is ready.

Table 16-9: Possible Settings for Bits 1 and 2, Status Bits for DPR Communication

Bit 4: Fieldbus Slave Ready

Status bit for the active bus capabilities of the fieldbus slave (FB Slave Ready). This bit is monitored for the Fieldbus Error Reaction. Whenever this bit goes to 0 after a fieldbus card was initially found by the PPC-R, the selected Error Reaction (system shutdown, error message, or ignore) is initiated.

0 = The fieldbus slave is not (yet) ready for data exchange.

1 = The fieldbus slave can actively participate on the bus.

Bit 5: Non Cyclic Ready

Status bit for the non-cyclic channel (Parameter Channel):

0 = The cyclic channel (Parameter Channel) cannot (yet) be used.

1 = The non-cyclic channel (Parameter Channel) is ready for use by the fieldbus master.

Bit 14: Register Data Valid (PLC Interface only)

Status bit for the register data output:

0 = The register channel data (from PLC to PPC) are invalid.

1 = The register channel data (from PLC to PPC) are valid.

Bit 15: Cyclic Data Valid

Status bit for the cyclic data output:

0 = The cyclic data outputs (coming in to the PPC-R) are INVALID.

1 = The cyclic data outputs (coming in to the PPC-R) are VALID. The system looks for this bit to be 1 before allowing data transfer.

Register 20: Fieldbus_PLC Diagnostics

Register 20 holds the information for "Fieldbus Diagnostics." Table 16-10 below contains the bit assignment for the diagnostic object 5ff0.

Register	Register Label Name	Bit	Function
20	Fieldbus/PLC Diagnostics	1	Not Used
		2	Not Used
		3	Not Used
		4	Not Used
		5	Not Used
		6	Not Used
		7	Not Used
		8	Not Used
		9-12	Not Used
		13	FB Type, LSB
		14	FB Type Bit 2
		15	FB Type Bit 3
		16	FB Type, MSB

Table 16-10: Register 20: Fieldbus Diagnostics

Bits 13- 16: Fieldbus/PLC Type

These bits identify the type of fieldbus interface card detected by Rexroth VisualMotion. The bit combinations for Bits 13, 14, 15 and 16 are as listed in the table below.

Bit 16	Bit 15	Bit 14	Bit 13	Fieldbus Type
0	0	0	0	<NO CARD>
0	0	0	1	<Not Defined>
0	0	1	0	Interbus
0	0	1	1	DeviceNet
0	1	0	0	Profibus
0	1	0	1	ControlNet
0	1	1	0	<Not Defined>
0	1	1	1	Ethernet/IP (10 MB and 100MB)
1	1	1	1	Rexroth PLC Interface

Table 16-11: Identification of the Fieldbus Interface

Register 21: System Status

Register 21 is a read-only register dedicated to system status. The status is indicated by a high level in the appropriate bit.

Register	Register Label Name	Bit	Function
21	System Status	1	Parameter Mode
		2	Alternate Sercos Communication
		3	Not Used
		4	Service Channel Ready
		5	Error
		6	Error Active
		7	Warning Active
		8	PLC Error Active
		9	Active Program, LSB
		10	Active Program Bit 2
		11	Active Program Bit 3
		12	Active Program, MSB
		13	TP Password Active
		14	Teach Pendant
		15	Flash Backup Active
		16	Autostore Copy Valid

Table 16-12: Register 21: System Status

Bit 1: Parameter Mode/ Initializing

0 = Run Mode, 1 = Parameter Mode or Initializing System

If this bit is (1), the control is in parameter mode or the system is being initialized into run mode. If parameter mode is selected in System Control bit 1, the drives are in phase 2, access to restricted parameters is allowed, and the user programs are stopped.

If this bit is (0), the control is in run mode, and the user program is ready for operation if there are no errors.

Bit 2: Alternate Sercos Communication

This bit is set to (1) when an alternative method of Sercos communications is used.

Note: This requires bootloader FWC-PPCPRs-BTL-02V06 to be used.

Bit 4: Service Channel Ready

This bit can be checked by a user interface before initiating communication with a drive. When set to (0), the Sercos ring is disconnected or phases are being switched. When set to (1), the drives are ready for service channel communication.

Bit 5: Error

This is the global error indicator for the Rexroth VisualMotion system. If the system, any task, or any drive has an error, this bit is set to (1). If there are no errors present, it is set to (0).

Bit 6: Error Active

This bit is set to (1) when the current diagnostic is Fatal or Non-Fatal.

Bit 7: Warning Active

This bit is set to (1) when the current diagnostic is a Warning.

Bit 8: PLC Error Active

This bit is set to (1) when a PLC error has been detected in the system.

Bits 9-12: Active Program

These bits represent the currently active program as a binary program number. (Bit 12= most significant bit)

Example The bits below indicate that program 5 is active.

Bit 9: 1	Bit 10: 0
Bit 11: 1	Bit 12: 0

Bit 13: Teach Pendant Password Active

This bit is set if the teach pendant password is active. It allows the user program or IndraLogic project to disable functions while the corresponding functions are disabled in the teach pendant.

Bit 14: Teach Pendant Connected

This bit is set if the teach pendant is connected.

Bit 15: Flash Backup Valid

This bit is set to (1) when a valid Compact flash backup is stored in the PFM module.

Bit 16: Autostore Copy Valid

This bit is set to (1) when the NVRam contents has been successfully stored in the PFM module and is current with the project. Any modifications to the file system makes the backup invalid.

Registers 22-25: Task Status

Registers 22, 23, 24 and 25 are dedicated to task status for Tasks A, B, C and D respectively. The condition of the task status registers may be read by a user task program to determine the state of the system and the task's program execution.

Register	Register Label Name	Bit	Function
22	Task A Status	1	Mode Automatic nManual
23	Task B Status	2	Coordinated Running
24	Task C Status	3	Not Used
25	Task D Status	4	Single Step
		5	Task Error
		6	Task Running
		7	Not Used
		8	Coordinate In Position
		9	Not Used
		10	Trace Ready
		11	Breakpoint Reached
		12 – 14	Not Used
		15	Coordinated Fast Stop
		16	Not Used

Table 16-13: Register 22-25: Task Status

Bit 1: Mode: Automatic/Manual

This bit is set to (1) by the control when the task is in automatic mode and is ready for the program to be started. It is set to (0) when manual mode is selected, an error is preventing the task from starting, or the task has not been initialized into automatic mode.

Bit 2: Coordinated Running

This bit is set to (1) by the control when coordinated motion is ongoing in the task. When motion stops, the bit is reset to 0.

Bit 4: Single Stepping

When this bit is set to (1), the program flow has been stopped after an instruction, sequencer function, or sequencer step in single-step mode. To resume program flow, a (0-1) transition on the cycle start bit is required.

Bit 5: Task Error

This bit is set to (1) by the control when a task has an error or warning condition. An error is shown in parameter T-0-0122. The bit is (0) when no errors exist in this task.

Bit 6: Task Running

This bit is set to (1) by the control when the user task is executing program instructions, either from the main task or from an event. It is (0) when the task is not running.

Bit 8: Coordinate In Position

This bit is set to (1) when a coordinate segment or path has reached its end position and stops.

Bit 11: Breakpoint Reached

When this bit is set to (1), the breakpoint enabled in task parameter T-0-0137, activated with Task Control Register bit 11, has been reached. Program flow has been stopped. To resume program flow, a low-to-high (0-1) transition on the cycle start bit is required.

Bit 15: Coordinated Fast Stop

This bit is set to (1) after a coordinated fast stop is performed and the task motion comes to a complete stop using task parameter values.

Register 27: Initialization Task Control

The register is reserved for future use as a control register for the Initialization task.

Register 28: Initialization Task Status

The register is reserved for future use as a control register for the Initialization task.

Registers 31-38, 309-340, 541-564: Axis Status

Registers 31-38 are for axes 1-8. Registers 309-340 (read only) for axes 9-40. Registers 541-564 for axes 41-64. These registers are reserved for control axis status. These axes correspond to the Sercos drive address on the fiber-optic communication loop.

Register	Register Label Name	Bit	Function
31-38 309-340 541-564	Axis 1-8 Axis 9-40 Axis 41-64	1	Multiplex Channel Enabled
		2	Jogging Forward
		3	Jogging Reverse
		4	Phase Adjusted
		5	ELS Enabled
		6	ELS Secondary Mode
		7	Axis In-Position
		8	Axis Aligned
		9	Not Used
		10	Axis Stopped / Axis Present on Ring
		11	Axis Halted
		12	Class 3 Status
		13	Class 2 Warning
		14	Shutdown Error
		15	Drive Ready LSB
		16	Drive Ready MSB

Table 16-14: Registers 31-38: Axis Status

Bit 1: Multiplex Channel Enabled

This status bit is set to (1) if the Sercos multiplex channel is enabled. The Sercos multiplex channel can be enabled by either the selection of the Drive PLS Fast Write feature (axis parameter A-0-0004 Axis Options bit 8) or the automatic detection of the AT or MDT exceeding the 16 byte limit of the DKC2.3 drive.

Bits 2 and 3: Jogging Forward (Bit 2) and Reverse (Bit 3)

This status bit is set to (1) when the axis is jogging forward (Bit 2) or reverse (Bit 3) through either the teach pendant or the I/O bits. Otherwise, it is set to (0).

Bit 4: Phase Adjusted

For ELS and Cam axes, this bit indicates if a phase adjustment move has been completed. During the phase adjust, it is set to (0), indicating the adjustment is in progress. When the phase adjust is complete, or when first synchronizing, this bit is set to (1).

Bit 5: ELS Enabled

When this bit is set to (1), the axis is in ELS or cam mode. When set to (0), it is in velocity or single-axis mode, as indicated by bit 6.

Bit 6: ELS Secondary Mode

For ELS or cam axes, this indicates the current secondary mode that is enabled when ELS/cam is disabled. If it is set to (0), the axis is in single-axis mode. If it is set to (1), the axis is in velocity mode.

Bit 7: Axis in Position (Single Axis)

Bit 7 is set to one (1) when the target position is reached and the axis is at zero velocity in single-axis mode. The drive associated with the axis must have its in-position window and zero velocity window parameters set correctly for this bit to function properly.

Bit 8: Axis Aligned (Control cam axes only)

This bit provides the status of control based cam alignment. It is set to (1) if the axis is aligned to the cam, and (0) if it is not aligned. This allows user program logic to determine if an alignment move or phase offset is needed. This bit is only for control cam axes. It is not checked for Drive Cams.

The following conditions set this bit to (0):

- The axis is not configured in the program to be a cam axis.
- A valid cam is not active for this axis.
- The absolute value of (position of the axis - slave position from cam equation) is greater than the in-position window (drive parameter S-0-0057).

The following conditions set this bit to (1):

- The axis is synchronized to the master
- The absolute value of (position of the axis - slave position from cam equation) is less than or equal to the in-position window (drive parameter S-0-0057).

Note: When monitoring this bit on IndraDrives, ensure that the relevant drive parameter P-0-0187 is set to 0. Otherwise, the effectiveness of this signal is diminished.

Bit 10: Axis Stopped / Axis Present on Ring

This bit corresponds to Drive Parameter S-0-0182, bit 1.

During runtime (phase 4), this bit is set to (1) when the feedback velocity is less than the drive's zero velocity window.

In Parameter mode, this bit is set to (1) to indicate that the axis is present on the drive’s Sercos ring.

Bit 11: Axis Halted

Bit 11 is set to one (1) when the drive's restart/Inhalt bit is set to zero (0) (drive halted) and the axis is at zero velocity in single-axis mode.

Bit 12: Class 3 Status

This bit corresponds to the Change in Class 3 Diagnostics bit in the drive status word. When the diagnostic condition changes, the drive changes this bit from a (0-1). The program can then check this register bit for a (0-1) transition, instead of continually reading the status parameter through the service channel.

Bit 13: Class 2 Warning

This bit indicates that a Class 2 warning condition exists. It is set (1) when a warning occurs and is not cleared (0) until the warning is cleared. Warnings are often temporary conditions. This bit allows the program to latch and take action on a warning.

Bit 14: Shutdown Error (Drive)

Bit 14 is set to one (1) when there is a Class 1 Diagnostic Shutdown Error in the drive. This bit corresponds to the same bit in the drive's Sercos status register.

Bit 15, 16: Ready to Operate

Bits 15 and 16 indicate when a drive is ready to operate. When both bits are one (1), the drive will respond to motion commands. These bits correspond to the drive's Sercos status register. ***See Drive Parameter S-0-0135.***

Bit 16	Bit 15	Description	Digital Drive LED Code
0	0	Drive not ready for power up	error or P1 - P3
0	1	Drive ready for power up	bb
1	0	Drive control and power sections ready	Ab
1	1	Drive ready to operate	AH or AF

Table 16-15: Description of Bits 15 and 16 for Axis Status

Registers 40: Link Ring Status

This Link Ring status register monitors errors in Link Ring cross communication and fiber optic rings (primary and secondary) for the configured Link Ring node.

Register	Register Label Name	Bit	Function
40	Link Ring Status	1-3	Not used
		4	Link Error
		5	Error Primary Optic Ring
		6	Error Secondary Optic Ring
		7	Redundancy Loss
		8-16	Not Used

Table 16-16: Registers 40: Link Ring Status

Bit 4: Link Error

The Link Error status bit is set when a Link Ring error currently exists. All "541 Link Ring Error, see ext. diag" errors will set this bit until cleared.

Bit 5: Error in Primary Optic Ring

This bit is set when an error occurs in the primary Link Ring. If using a double Link Ring, the Redundancy_Loss bit in all active Link Ring nodes is set to indicate that additional Link Ring errors may result in Link Ring failure.

Bit 6: Error in Secondary Optic Ring

This bit is set when an error occurs in the secondary Link Ring. Although an error in the secondary ring may not affect signal transmission in the primary Link Ring, the Redundancy_Loss bit in all active Link Ring nodes is set to indicate that redundancy is no longer available and that an error in the primary Link Ring may result in Link Ring failure. This bit is only relevant for double Link Rings.

Bit 7: Redundancy Loss

This bit is set when the redundancy provided by the secondary Link Ring is no longer available. Loss of redundancy can occur when there is an error in the primary, secondary, or both rings. The Redundancy_Loss bit is set simultaneously in all nodes.

Registers 41, 42: Link Ring Data 1

These registers monitor signal validity and transmission across the Link Ring for active nodes 1–16 (Register 41) and nodes 17–32 (Register 42). Each bit in these registers corresponds to an active node in the Link Ring. The associated bit is deactivated if the master position generated by the node is not valid or the node is not transmitting the data to the Link Ring.

Register	Register Label Name	Bit	Function
41	Link Ring Data 1	1-16	Nodes 1-16 Data Valid
42	Link Ring Data 2	1-16	Nodes 17-32 Data Valid

Table 16-17: Registers 41, 42: Link Ring Data 1

Bits 1-32: Nodes 1-32 Data Valid

When set to (1), the node is transmitting valid ELS Master position data across the Link Ring in Sercos phase 4.

When set to (0), the node is no longer transmitting valid data across the Link Ring or the node is no longer is Sercos phase 4.

These bits can also be used to trigger an error reaction, such as switching the group to a local master other than the Link Ring Master or switching the control to local mode. To trigger an error reaction, map the bits in an IndraLogic project where you can create logic to execute the error reaction.

Register 44: PPC Onboard XI

This register is mapped to the digital inputs 1, 2, and 3, found on the X1 connector of the control. Bits 1, 2, and 3 are mapped according to the following table:

Register	Register Label Name	Bit	Function
44	PPC_Onboard_XI	1	Mapped to pin 3 on connector X1
		2	Mapped to pin 4 on connector X1
		3	Mapped to pin 5 on connector X1

Table 16-18: Register 44: PPC Onboard XI

Register 45: PPC Onboard XO

This register is mapped to the digital outputs 1 and 2, found on the X1 connector of the control. Bits 1 and 2 are mapped according to the following table:

Register	Register Label Name	Bit	Function
45	PPC_Onboard_XO	1	Mapped to pin 1 on connector X1
		2	Mapped to pin 2 on connector X1

Table 16-19: Register 45: PPC Onboard XO

Register 50: Ethernet 1 Status

This register monitors the status of the network communication of either the on-board Ethernet interface or the Ethernet Card, depending on the current hardware configuration. The following hardware conditions should be taken into consideration:

- If an Ethernet card interface is installed in the PPC-R22.1 control, then this register is assigned to the card.
- If no Ethernet card is installed but an on-board Ethernet interface is installed, then this register is assigned to the on-board Ethernet interface.
- If both interfaces are installed, then this register is assigned to the Ethernet card and register 55 is assigned to the on-board Ethernet interface.

Register	Register Label Name	Bit	Function
50	Ethernet 1 Status	1	Card Present
		2-8	Not Used
		9	Request Received
		10	Response Pending
		11	Response Done
		12	Response Sent
		13-15	Not Used
		16	Invalid Protocol

Table 16-20: Registers 50: Ethernet 1 Status

Bit 1: Card Present

This bit monitors the presence of the Ethernet Interface.

0 = Ethernet interface is ***not*** present.

1 = Ethernet interface is present

Bit 9: Request Received

A (1) indicates that a request has been received and that the CIF driver function is executing.

Bit 10: Response Pending

A (1) indicates that the message received is currently being processed by the control. The response is pending on the completion of the message.

Bit 11: Response Done

A (1) indicates that the message has been processed by the control and is complete.

Bit 12: Response Sent

A (1) indicates that the response message has been sent to the DDE Server.

Bit 16: Invalid Protocol

A (1) indicates that an invalid protocol has been received (standard or encrypted ASCII).

Register 51: Standard Message Count

This register indicates the number of messages that have been received in standard ASCII protocol. The counter is incremented for any ASCII message received from any device that can communicate the ASCII protocol to the control. An ASCII message must start with a ‘>’ and follow the standard ASCII protocol.

This register is initialized to zero on power up. Therefore, the counter will start incrementing once messages are received after the control has been powered up. If the counter is incremented, a message is processed by the control and returned to the sender. The maximum number of messages that can be counted is 65535 before the counter is reset.

Register 52: Cyphered Message Count

This register indicates the number of messages that have been received in encrypted ASCII protocol. This counter will only be incremented for the Ethernet because the encrypted ASCII protocol is only used with Ethernet. Rexroth VisualMotion does not support encryption of SIS protocol messages over Ethernet.

This registers is initialized to zero on power up. Therefore, the counter will start incrementing once messages are received after the control has been powered up. If the counter is incremented, a message is processed by the control and returned to the sender. The maximum number of messages that can be counted is 65535 before the counter is reset.

Register 53: Invalid Protocol Count

This register indicates the number of messages that have been received with a protocol that can not be determined. Any device that attempts to communicate in an unknown communication protocol will cause this counter to increment. This can occur when the start character of a message did not start with a ‘>’ (ASCII protocol) or a ‘02” (SIS protocol). This register is also incremented if an invalid SIS message is received. In this case the control will respond with the proper SIS error message. For all other cases, the control will not respond with an error, rather it will ignore all characters received and increment this counter accordingly.

Hardware errors will not cause this counter to increment. This counter will only increment if there has been no error detected from the device hardware and an invalid unrecognized message is received.

This registers is initialized to zero on power up. The maximum number of messages that can be counted is 65535 before the counter is reset and will start counting from zero again.

Register 54: SIS Message Count

This register indicates the number of messages that have been received in the SIS protocol. A SIS message must start with a ‘02’ and follow the standard SIS protocol. This counter is only incremented if a valid SIS message is received by the control. All valid messages from any device that can communicate the SIS protocol to the control will increment the counter. If an invalid SIS message is received then register 53 will be incremented.

This registers is initialized to zero on power up. Therefore, the counter will start incrementing once messages are received after the control has been powered up. If the counter is incremented, a message is processed by the control and returned to the sender. The maximum number of messages that can be counted is 65535 before the counter is reset.

Registers 55: Ethernet 2 Status

This register monitors the status of the network communication of the on-board Ethernet interface when both the Ethernet card and on-board Ethernet interfaces are installed in a PPC-R22.1 control.

Register	Register Label Name	Bit	Function
55	Ethernet 2 Status	1	Card Present
		2-8	Not Used
		9	Request Received
		10	Response Pending
		11	Response Done
		12	Response Sent
		13-15	Not Used
		16	Invalid Protocol

Table 16-21: Registers 55: Ethernet 2 Status

Bit 1: Card Present

This bit monitors the presence of the Ethernet Interface.

0 = Ethernet interface is ***not*** present

1 = Ethernet interface is present

Bit 9: Request Received

A (1) indicates that a request has been received and that the CIF driver function is executing.

Bit 10: Response Pending

A (1) indicates that the message received is currently being processed by the control. The response is pending on the completion of the message.

Bit 11: Response Done

A (1) indicates that the message has been processed by the control and is complete.

Bit 12: Response Sent

A (1) indicates that the response message has been sent to the DDE Server.

Bit 16: Invalid Protocol

A (1) indicates that an invalid protocol has been received (standard or encrypted ASCII).

Registers 86: PMG Control

This register is used to enable the Position Monitoring Groups and set an offset for deviation monitoring between axis in a group.

Register	Register Label Name	Bit	Function
86	PMG Control	1-8	Enables Position Monitoring Group
		9-16	Calculates necessary offset of axis in groups 1-8 respectively to achieve 0 deviation.

Table 16-22: Registers 86: PMG Control

Bit 1-8: PMG#_ENABLE

These bits are used to enable the Position Monitoring feature for groups 1-8 respectively. A (0 ⇒ 1) transition, enables the PMG for the selected group.

Bit 9-16: PMG#_CALC_OFFSET

These bits are used to calculate the necessary position offset to achieve a 0 deviation between slave axes and the primary axis in groups 1-8, respectively. The calculated offset values are written to the following control parameters:

- C-0-3203 for group 1
- C-0-3213 for group 2
- C-0-3223 for group 3
- C-0-3233 for group 4
- C-0-3243 for group 5
- C-0-3253 for group 6
- C-0-3263 for group 7
- C-0-3273 for group 8

A (0 ⇒ 1) transition of these bits calculates the offset difference between the assigned slave axes and the primary axis. This value is stored in RAM memory until the PMG_ENABLE bit is set for the same group.

Note: If an offset value is modified, the PGM_ENABLE bit for the group must be set again in order for the offset to take affect.

Registers 87: PMG Status

This register displays the status of the PMG#_ENABLE and PMG#_ERROR condition.

Register	Register Label Name	Bit	Function
87	PMG Status	1-8	Position Monitoring Group Enabled
		9-16	Error reaction indication

Table 16-23: Registers 87: PMG Status

Bit 1-8: PMG#_ENABLED

These bits display the status of the PMG_ENABLE bits of control register 86. When set to 1, the Position Monitoring feature for the selected group has been enabled.

Bit 9-16: PMG#_ERROR

When set to 1, an error was generated from the Position Monitoring Group feature.

Registers 88 and 89: Task A Extend Event Control

Register 88 (USER_XI_REG) is used to trigger up to 16 events in Task A. Register 89 (USER_XO_REG) is used to monitor the status of events triggered by Register 88.

Together they provide a time critical way to control motion on the control without time wasting polling loops. These events are similar to the Task Input Transition, located in each task control register, but are limited to the task with the highest priority (Task A).

Register Operation

At power up and at system reset the output register USER_XO_REG is cleared.

The input register USER_XI_REG triggers an EUI on positive edge bit transitions. The event function assigned to the bit is executed at that time. Only positive going edges in USER_XI_REG can trigger an EUI.

The output register USER_XO_REG is effected by the Event Setup Box Icon arm and disarm functions as well as by bit transitions within USER_XI_REG.

When an EUI is armed through the Event Setup Box Icon, its bit is set in USER_XO_REG. This provides an external output that indicates that the EUI is armed and ready for operation. Likewise, it is reset if disarmed through the icon.

When an EUI is triggered, its corresponding output bit in USER_XO_REG is reset (normally it is set) providing an external output indicating that the event is active. It remains reset so long as no high-to-low (1-0) transition on the corresponding input bit in USER_XI_REG detected. When a high-to-low (1-0) transition on the input bit is detected in USER_XI_REG, the output bit in USER_XO_REG is again set indicating the event is armed and ready for another positive going edge.

Registers 90 and 91: Latch and Unlatch

Register 90 provides 16 latches that can be set to (1) directly or in an IndraLogic project. The bits in register 90 can be reset to (0) only by writing a (1) to the corresponding bit in register 91. Register 91 immediately clears its bits to (0), so that a transition is not needed on the unlatch bits.

Registers 92-94: Mask Pendant Key Functionality

The bits in registers 92-94 “mask” the functionality of the corresponding bits in registers 95-97. This means that if a bit is set to 1 (on) in register 92, 93 or 94, the BTC06 key controlled by the corresponding bit in register 95, 96 or 97 is not operational. (Register 92 masks register 95, register 93 masks register 96 and register 94 masks register 97.) This masking feature allows the programmer to redefine the action of a key or prevent the action normally mapped to that key.

Examples

Register 92, bit 1 is set to 1. Pressing the F1 key (status given in register 95, bit 1) does not result in the expected action.

Register 93, bit 7 is set to 1. Pressing the X+ key (status given in register 96, bit 7) does not result in jogging in the Jog Menu.

Note: When a key on the BTC06 is masked, a key press still sends an acknowledgement to the status register bit, although no action occurs. This means that the key can be mapped to perform a different function, or the usual function can be prevented.

Note: If an F-key is masked off, the action assigned to it will not appear at the bottom of the BTC06 screen of the Control Menu.

Registers 95-97: BTC06 Teach Pendant Status

The bits in these registers are set when the corresponding keys are pressed on the teach pendant. They can be scanned in the user program or by IndraLogic to detect system operations or to extend teach pendant functionality to control the user program.

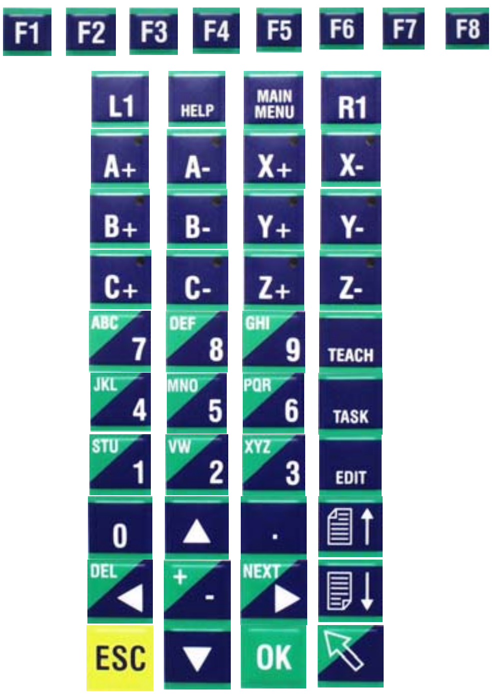
Note: These bits are all read-only.

The BTC06 keyboard is mapped to register 95, 96 and 97. The figure below and to the right outlines the register and bit location in the following format:

Register - Bit

Example: **95 – 01,** *key is mapped to register 95, bit 01*

When a key is pressed its corresponding bit turns on and remains on for as long as the key is pressed.



95-01	95-02	95-03	95-04	95-05	95-06	95-07	95-08

97-13	95-09	97-14	97-15
96-05	96-06	96-07	96-08
96-13	96-14	96-15	96-16
97-05	97-06	97-07	97-08
95-10	95-11	95-12	95-13
96-02	96-03	96-04	95-14
96-10	96-11	96-12	96-09
97-03	95-11	97-02	95-15
96-02	97-04	96-04	95-16
96-01	96-11	97-01	

This shift key is not mapped to a register.

Fig. 16-2: Teach Pendant: Function Keys to Register Map

Registers 98: Pendant Control - Task A, B

The bits in this register can disable teach pendant control of the selected function for the corresponding Tasks A-B.

Register	Register Label Name	Bit	Function
98	Pendant Control – Task A-B	1	Block Task A Manual
		2	Block Task A Auto
		3	Block Task A Step
		4	Block Task A Jog
		5	Block Task A Entry
		6	Block Task A Teach
		7 and 8	Not Used
		9	Block Task B Manual
		10	Block Task B Auto
		11	Block Task B Step
		12	Block Task B Jog
		13	Block Task B Entry
		14	Block Task B Teach
		15 and 16	Not Used

Table 16-24: Register 98: Pendant Control – Task A-B

Registers 99: Pendant Control - Task C-D

The bits in this register can disable teach pendant control of the selected function for the corresponding Tasks C-D.

Register	Register Label Name	Bit	Function
99	Pendant Control – Task C-D	1	Block Task C Manual
		2	Block Task C Auto
		3	Block Task C Step
		4	Block Task C Jog
		5	Block Task C Entry
		6	Block Task C Teach
		7 and 8	Not Used
		9	Block Task D Manual
		10	Block Task D Auto
		11	Block Task D Step
		12	Block Task D Jog
		13	Block Task D Entry
		14	Block Task D Teach
		15 and 16	Not Used

Table 16-25: Register 99: Pendant Control – Task C-D

Register 100: IO Box Inputs

This is the default register used for the input bits of the VisualMotion I/O Box utility. This register is only active when a valid I/O mapping project is downloaded to the control from IndraLogic. Refer to section 4.2, *Import Default IndraLogic Project*, in volume 1 of the *VisualMotion 11 Functional Description* for details. The bits are assigned to the IO Box according to the following table.

Register	Register Label Name	Bit	Function
100	IO Box Inputs	1	Parameter Mode
		2	Automatic Mode
		3	Stop
		4	Start
		5	Clear Error
		6	Live-Man
		7	Emergency Stop
		8	Activate Program
		9-16	not used

Table 16-26: Register 100: IO Box Inputs

When register 100 is assigned to the VisualMotion I/O Box inputs, the bits in register 1 (System Control) and registers 2-5 (Task Control A-D) are mapped to register 100 according to the following table:

Register 100 Bit	Register 1 Bit	Registers 2-5 Bits	Description
1	1	--	Parameter Mode
2	--	2	Automatic Mode
3	--	7	nTask Stop
4	--	6	Cycle Start
5	--	5	Clear Error
6	6	--	Live-Man
7	3	--	nE-Stop
8	8	--	Activate Program

Table 16-27: Mapping of Register 100

Register 120: IO Box Outputs

This is the default register used for the output bits of the VisualMotion I/O Box utility. This register is only active when a valid I/O mapping project is downloaded to the control from IndraLogic. Refer to section 4.2, *Import Default IndraLogic Project*, in volume 1 of the *VisualMotion 11 Functional Description* for details. The bits are assigned to the IO Box according to the following table.

Register	Register Label Name	Bit	Function
120	IO Box Inputs	1	Parameter Mode
		2	Automatic Mode
		3-16	not used

Table 16-28: Register 100: IO Box Inputs

When register 120 is assigned to the VisualMotion I/O Box outputs, the bits in register 21 (System Status) are mapped to register 120 according to the following table:

Register 120 Bit	Register 21 Bit	Description
1	1	Parameter Mode
2	5	Error

Table 16-29: Mapping of Register 120

Register 140: ELS Master Control

This register is used to enable slip monitoring between two ELS System Masters. Slip monitoring is configured in the *Slip Monitor Setup* window that is activated by selecting the ***Slip Monitor Setup…*** button in the ELS System Master icon.

Note: This register number can be assigned by Rexroth VisualMotion as the default system register when creating an icon program or can be changed to any number not reserved by Rexroth VisualMotion. It is strongly recommend that the programmer use the default number. This makes documentation and modification to user programs an easier task over the scope of the project.

Register	Register Label Name	Bit	Function
140	ELS Master Control	1-6	Reserved
		7	Set ELS Master 1 Reference Position
		8	Set ELS Master 2 Reference Position
		9	Set ELS Master 3 Reference Position
		10	Set ELS Master 4 Reference Position
		11	Set ELS Master 5 Reference Position
		12	Set ELS Master 6 Reference Position
		13-14	Reserved
		15	Capture Slip Monitoring
		16	Enable Slip Monitoring

Table 16-30: Register 140: ELS Mater Control

Bits 7- 12: Set ELS Master (1-6) Reference Position

When a secondary feedback device is used as an ELS Master (Real Master), it is sometimes necessary to reference the ELS Master output position (ELS_MSTR_POS#) using the ELS Master reference position (ELS_MSTR_REF_POS#). A low-to-high (0-1) transition of this bit, while in phase 4, copies the ELS Master reference position to the ELS Master output position.

Note: To ensure accuracy and avoid any sudden movement of an ELS Group, following the ELS Master, this command should only be activated when no motion is present.

Bit 15: Capture Slip Monitoring

This bit is used to capture the value of the maximum allowed deviation window variable (ELS_MSTR_SLIP_WINDOW), and the master position offset variable (ELS_MSTR_SLIP_OFFSET) when two ELS System Masters are configured for slip monitoring. The value of ELS_MSTR_SLIP_OFFSET is calculated if the **Master Position Offset** selection is set to **Dynamically reset phase offset on system phase up** in the *Slip Monitor Setup* window. This is the default setting.

A low-to-high (0-1) transition of this bit captures and writes the maximum allowed deviation window and master position offset variable values to the control's memory during run-time.

Note: If the ELS_MSTR_SLIP_WINDOW variable is modified from its initial value, this bit must transitioned from 0-1 for the new value to take effect.

Bit 16: Enable Slip Monitoring

This bit enables or disables slip monitoring between two ELS System Masters.

0 = Disable slip monitoring

1 = Enable slip monitoring

Register 141 ELS Master Status

This register monitors the maximum position difference using two master encoder signals.

Note: This register number can be assigned by Rexroth VisualMotion as the default system register when creating an icon program or can be changed to any number not reserved by Rexroth VisualMotion. It is strongly recommend that the programmer use the default number. This makes documentation and modification to user programs an easier task over the scope of the project.

Register	Register Label Name	Bit	Function
141	ELS Master Status	1	Master 1 at Standstill
		2	Master 2 at Standstill
		3	Master 3 at Standstill
		4	Master 4 at Standstill
		5	Master 5 at Standstill
		6	Master 6 at Standstill
		7	ELS Master 1 Position Referenced
		8	ELS Master 2 Position Referenced
		9	ELS Master 3 Position Referenced
		10	ELS Master 4 Position Referenced
		11	ELS Master 5 Position Referenced
		12	ELS Master 6 Position Referenced
		13	Reserved
		14	Monitoring ERROR Active
		15	Lead Encoder
		16	Slip Monitoring Enabled

Table 16-31: Register 141: ELS Master Status

Bit 1-6: Master at Standstill

The *Master at Standstill* bit is set to 1 when the master’s output velocity value is at or below the Slip Masters Velocity Threshold variable (ELS_MSTR_STANSTILL) for two Sercos cycles.

Bits 7- 8: ELS Master (1-6) Position Referenced

This bit is set to 1 when the ELS Master reference position has been copied to the ELS Master output position. The set command is activated using control register 140, ELS Master Control, bits 1-6.

0 = ELS Master **not** referenced

1 = ELS Master referenced

Bit 14: Monitoring ERROR Active

This bit is set to 1 when a slip monitoring error is encountered. Perform the following steps to clear this error:

1. Disable slip monitoring (ELS Master control register, bit 16 = 0).
2. Transition the Clear All Error bit from 0-1(Register 001, Bit 5).

Bit 15: Lead Encoder

This bit is used when the primary and secondary masters used in slip monitoring are Real Masters. Its used to indicate which master caused a slip monitoring error.

0 = Primary Master

1 = Secondary Master

Bit 16: Enable Slip Monitoring

This bit monitors the status of the Enable Slip Monitoring bit.

0 = Slip monitoring disabled

1 = Slip monitoring enabled

Registers 150 and 151: Virtual Master 1 & 2 Control

GPP 11 supports two Virtual Masters. A Virtual Master is an internal motion engine with an independent set of control parameters. Each Virtual Master can be used independently from each other.

A Virtual Master is controlled by the Rexroth VisualMotion user program created with Rexroth VisualMotion Toolkit (VMT) and/or a PLC using I/O registers and program variables. The initialization of these registers and program variables is defined in the Virtual Master icon.

Note: These register numbers can be assigned by Rexroth VisualMotion as default system registers when creating an icon program or can be changed to any register number not reserved by Rexroth VisualMotion. It is strongly recommend that the programmer use the default numbers. This makes documentation and modification to user programs an easier task over the scope of the project.

Register	Register Label Name	Bit	Function
150 and 151	Virtual Master 1and 2 Control	1	VM#_CT_FSTOP
		2	VM#_CT_HOME
		3	VM#_CT_GO
		4	VM#_CT_VMODE
		5	VM#_CT_RELMODE
		6	VM#_CT_RELTRIG
		7-16	Not Used

Table 16-32: Registers 150-151: Virtual Master 1 & 2 Control

Bit 1: Virtual Master 1or 2 Fast Stop

A low-to-high (0-1) transition decelerates the Virtual Master using the programmed E-Stop deceleration variable (VM#_E_STOP_DECEL). No motion is possible while this bit remains high (1).

Bit 2: Virtual Master Home Position

A low-to-high (0-1) transition returns the Virtual Master to the programmed home position in the VM#_HOME_POS variable.

Bit 3: Virtual Master Go Command

A low-to-high (0-1) transition starts the selected Virtual Master. A high-to-low (1-0) transition stops the Virtual Master.

Bit 4: Virtual Master Mode of Operation

Each Virtual Master supports two modes of operation:

0 = Positioning Mode

1 = Velocity Mode

Positioning Mode

Relative Positioning

A relative positioning move of a Virtual Master is used when a specific distance is required relative to the Virtual Master's current stop position variable VM#_CUR_POS. This distance (VM#_REL_MOVE_DIST) can be less than or greater than modulo (0-360 degrees). For example, if an indexing move requires that the Virtual Master travel 100 degrees from a stopped position of 350 degrees, enter a value of 100 for VM#_REL_MOVE_DIST and set bit 6 (VM#_CT_RELTRIG) to 1.

Note: A negative value for VM#_REL_MOVE_DIST will move the Virtual Master counter clockwise.

The following bit state will increment the current position of the Virtual Master by the value in variable VM#_REL_MOVE_DIST for every low-to-high transition of bit 6.

Bit 3 VM#_CT_GO	**Bit 5 VM#_CT_RELMODE**	**Bit 6 VM#_CT_RELTRIG**
1	1	0 ⇒ 1

Table 16-33: Relative Positioning for Virtual Master

If during a relative move, bit 3 (VM#_CT_GO) is switched to low (1-0), the relative move will stop. The move will be continued with a low-to-high (0-1) transition of bit 3 (VM#_CT_GO).

Absolute Positioning

Absolute positioning is used to position the Virtual Master between 0 and 360 degrees. The maximum travel distance with absolute positioning is 180 degree using shortest path or 359.9999 degree for positive or negative direction.

When bit 5 (VM#_CT_RELMODE) is set to 0, bit 3 (VM#_CT_GO) is set to 1 and bit 4 (VM#_CT_VMODE) is set to 0, the absolute target position in variable VM#_CMD_ABS_POS is used and every change of the target position will immediately trigger the positioning movement. If during an absolute move, bit 3 (VM#_CT_GO) is switched to 0 (1-0), the absolute move will be stopped. The move will be continued with a low-to-high (0-1) transition of bit 3 (VM#_CT_GO).

Velocity Mode

In velocity mode, the Virtual Master accelerates to the programmed value in velocity variable (VM#_CMD_VEL). A negative value in the velocity variable will cause the Virtual Master to move in the opposite direction. Additionally, it can stop at a predetermined stop position between 0 and 360 degree set by the VM#_STOP_POS variable.

- When bit 4 (VM#_CT_VMODE) is set to 1, a low-to-high (0-1) transition of bit 3 (VM#_CT_GO) will trigger the Virtual Master to accelerate to the programmed velocity variable (VM#_CMD_VEL). A high-to-low (1-0) transition of bit 3 (VM#_CT_GO) will cause an immediate stop using the programmed deceleration variable (VM#_CMD_DEC).

- A high-to-low (1-0) transition of bit 4 (VM#_CT_VMODE) with bit 5 (VM#_CT_RELMODE) set to 0 and bit 3 (VM#_CT_GO) set to 1 will cause a stop at the stop position variable (VM#_STOP_POS). VM#_STOP_POS will then overwrite VM#_CMD_ABS_POS.
- A high-to-low (1-0) transition of bit 4 (VM#_CT_VMODE) with bit 5 (VM#_CT_RELMODE) set to 0 and bit 3 (VM#_CT_GO) set to 0 will initialize the absolute positioning mode to the current position. It will replace the value in variable VM#_CMD_ABS_POS with the value of variable VM#_STOP_POS. A subsequent transition from 0-1 of bit 3 (VM#_CT_GO) will complete the programmed move of variable VM#_CMD_ABS_POS.
- A high-to-low (1-0) transition of bit 4 (VM#_CT_VMODE) with bit 5 (VM#_CT_RELMODE) set to 1 will cause an immediate stop with the programmed deceleration variable VM#_CMD_DEC (random stop position). In this case, the state of VM#_CT_GO does not matter.

Bit 5: Virtual Master Absolute/Relative Mode

Absolute or relative position mode of the Virtual Master is set with this bit. Positioning mode of a Virtual Master is set with the following combination of bits 4 and 5.

Active Position Mode	Bit 4	Bit 5
Absolute Mode	0	0
Relative Mode	0	1

Table 16-34: Setting Virtual Master Position Mode

Bit 6: Virtual Master Relative Trigger Mode

A low-to-high (0-1) transition triggers the relative move bit. This bit is only functional when a relative mode setup is configured using bits 4 and 5. The Virtual Master will move in increments of the value in variable VM#_REL_MOVE_DIST every time this bit transitions from low-to-high.

Registers 152-159: ELS Groups 1-8 Control

Note: These register numbers can be assigned by Rexroth VisualMotion as default system registers when creating an icon program or can be changed to any register number not reserved by Rexroth VisualMotion. It is strongly recommend that the programmer use the default numbers. This makes documentation and modification to user programs an easier task over the scope of the project.

Register	Register Label Name	Bit	Function
152-159	ELS Group 1-8 Control	1	G#_CT_LOCK_OFF
		2	G#_CT_M_REL_PH
		3	G#_CT_S_REL_PH
		4	G#_CT_MSTR_SEL
		5	G#_CT_VAR_CLK
		6	G#_CT_LOCAL
		7	G#_CT_LM_FSTOP
		8	G#_CT_LM_HOME
		9	G#_CT_LM_GO
		10	G#_CT_LM_VMODE
		11	G#_CT_M_ABS_PH
		12	G#_CT_S_ABS_PH
		13	G#_CT_LM_RELMODE
		14	G#_CT_LM_RELTRIG
		15	G#_CT_MSTR_FOL_PH
		16	G#_CT_MSTR_ABORT_PH

Table 16-35: Registers 152-159: ELS Group Control

Bit 1: Group Lock Off/ Lock on Cams

A low-to-high (0-1) transition actives the lock off cam profile. This cam profile decelerates to a stop over one cycle of the master. If the group's master requires additional cycles to stop, the cam's profile H factor is set to zero.

A high-to-low (1-0) transition actives the lock on cam profile. This profile accelerates from a stopped position to match the master's velocity over one cycle of the master (360 degrees). Once the velocity is matched, a one-to-one cam profile is active and follows the master.

Bit 2: Group Master Relative Phase Adjust

A low-to-high (0-1) transition of this bit adds a relative offset, prior to an optional group CAM profile, to the group’s output position. The relative move value is taken from program variable G#_PROG_M_PH. Each additional master phase adjust is added to the prior. The total or absolute master phase adjust, from the master input position, is written to program variable G#_ABS_M_PH.

Bit 3: Group Slave Relative Phase Adjust

A low-to-high (0-1) transition of this bit adds a relative offset, after an optional group master phase adjust and CAM profile, to the group's output position. The relative move value is taken from program variable G#_PROG_S_PH. Each additional slave phase adjust is added to the prior. The total or absolute slave phase adjust, from the master input position, is written to program variable G#_ABS_S_PH.

Bit 4: Group Master Input Select

This bit allows the ELS Master to be switched between the two configured masters (Virtual, Real or Group). The changeover can be done while the master is running.

When set to 0, the ELS Group and slave axes follow master 1 as selected in the user program and in the integer block variable (G#_MSTR1_AXIS).

When set to 1, the ELS Group and slave axes follow master 2 as selected in the user program and in integer block variable (G#_MSTR2_AXIS).

Bit 5: Group Forcing

When an ELS group is switched to local mode and the group master is at standstill, the internal variables of the group can be forced to a user defined value with a low-to-high (0-1) transition input of this forcing bit. The internal variables that can be changed through forcing are:

- internal group input master position
- group cam table input position
- ELS Group master position (only when bit 9 in ELS group configuration word is set to 1)
- state of the state machine for lock on / lock off
- absolute group master and group slave offset
- lock on, lock off and 1:1 cam profile ID number
- H-factor for lock on, lock off and 1:1 cam profile
- M and N factor

Bit 9 of the ELS Group configuration has influence on how forcing is done. The completion of forcing is indicated with bit 5, (forcing completed), of the group's status register.

Bit 6: Group Local Mode Select

A low-to-high (0-1) transition switches the group into local mode. First, the group's input master will be stopped using a stop ramp deceleration variable (G#_STOP_DECEL). After the group's input master is at standstill, the group can be jogged using bits 7-10 (group's jog engine) or group variables can be forced to a different value.

Bit 6, Group # Local Mode Status, of the group's status register indicates that forcing or jogging is possible.

A high-to-low (1-0) transition switches the group back to the selected group input master.

Bit 7: Group Local Virtual Master Fast Stop

A low-to-high (0-1) transition decelerates the Group Local Virtual Master using the G#_LM_E_STOP_DECEL programmed E-Stop deceleration variable. No motion is possible while this bit remains high (1).

Bit 8: Group Local Virtual Master Home Position

A low-to-high (0-1) transition returns the Group Local Virtual Master to the programmed home position in the G#_LM_HOME_POS variable.

Bit 9: Group Local Virtual Master Go Command

A low-to-high (0-1) transition starts the Group Local Virtual Master. A high-to low (1-0) transition stops the Group Local Virtual Master.

Bit 10: Group Local Virtual Master Mode of Operation

Each Virtual Master supports two modes of operation:

0 = Positioning Mode

1 = Velocity Mode

Positioning Mode

Relative Positioning

A relative positioning move of a Group Local Virtual Master is used when a specific distance is required relative to the Group Local Virtual Master's current stop position variable G#_LM_CUR_POS. This distance (G#_LM_REL_MOVE_DIST) can be less than or greater than modulo (0-360 degrees). For example, if an indexing move requires that the Group Local Virtual Master travel 100 degrees from a stopped position of 350 degrees, enter a value of 100 for G#_LM_REL_MOVE_DIST and set bit 14 (G#_CT_LM_RELTRIG) to 1.

Note: A negative value for G#_LM_REL_MOVE_DIST will move the Group Local Virtual Master counter clockwise.

The following bit state will increment the current position of the Group Local Virtual Master by the value in variable G#_LM_REL_MOVE_DIST for every low-to high transition of bit 14.

Bit 9 **G#_CT_LM_GO**	**Bit 13** **G#_CT_LM_RELMODE**	**Bit 14** **G#_CT_LM_RELTRIG**
1	1	0 ⇒ 1

Table 16-36: Relative Positioning for Group Local Virtual Master

If during a relative move, bit 9 (G#_CT_LM_GO) is switched to low (1-0), the relative move will stop. The move will be continued with a low-to-high (0- transition of bit 3 (VM#_CT_GO).

Absolute Positioning

Absolute positioning is used to position the Group Local Virtual Master between 0 and 360 degrees. The maximum travel distance with absolute positioning is 180 degree using shortest path or 359.9999 degree for positive or negative direction.

When bit 13 (G#_CT_LM_RELMODE) is set to 0, bit 9 (G#_CT_LM_GO) is set to 1 and bit 10 (G#_CT_LM_VMODE) is set to 0, the absolute target position in variable G#_LM_CMD_ABS_POS is used and every change of the target position will immediately trigger the positioning movement. If during an absolute move, bit 9 (G#_CT_LM_GO) is switched to 0 (1-0), the absolute move will be stopped. The move will be continued with a low-to-high (0-1) transition of bit 9 (G#_CT_LM_GO).

Velocity Mode

In velocity mode, the Group Local Virtual Master accelerates to the programmed value in velocity variable (G#_LM_CMD_VEL). A negative value in the velocity variable will cause the Group Local Virtual Master to move in the opposite direction. Additionally, it can stop at a predetermined stop position between 0 and 360 degrees set by the G#_LM_STOP_POS variable.

- When bit 10 (G#_CT_LM_VMODE) is set to 1, a low-to-high (0-1) transition of bit 9 (G#_CT_LM_GO) will trigger the Group Local Virtual Master to accelerate to the programmed velocity variable

(G#_LM_CMD_VEL). A high-to-low (1-0) transition of bit 9 (G#_CT_LM_GO) will cause an immediate stop using the programmed deceleration variable (G#_LM_CMD_DEC).

- A high-to-low (1-0) transition of bit 10 (G#_CT_LM_VMODE) with bit 13 (G#_CT_LM_RELMODE) set to 0 and bit 9 (G#_CT_LM_GO) set to 1 will cause a stop at the stop position variable (G#_LM_STOP_POS). G#_LM_STOP_POS will then overwrite G#_LM_CMD_ABS_POS.
- A high-to-low (1-0) transition of bit 10 (G#_CT_LM_VMODE) with bit 13 (G#_CT_LM_RELMODE) set to 0 and bit 9 (G#_CT_LM_GO) set to 0 will initialize the absolute positioning mode to the current position. It will replace the value in variable G#_LM_CMD_ABS_POS with the value of variable G#_LM_STOP_POS. A subsequent transition from 0-1 of bit 9 (G#_CT_LM_GO) will complete the programmed move of variable G#_LM_CMD_ABS_POS.
- A high-to-low (1-0) transition of bit 10 (G#_CT_LM_VMODE) with bit 13 (G#_CT_LM_RELMODE) set to 1 will cause an immediate stop with the programmed deceleration variable G#_LM_CMD_DEC (random stop position). In this case, the state of G#_CT_LM_GO does not matter.

Bit 11: Group Master Absolute Phase Adjust

A low-to-high (0-1) transition of this bit creates an absolute offset between the Group's gear ratio and optional CAM profile. The offset value is taken from program variable G#_PROG_M_PH. Additional transitions of this bit will not trigger another phase adjust unless the value in G#_PROG_M_PH is modified.

The value in program variable G#_PROG_M_PH is written to G#_ABS_M_PH after the phase adjust is complete.

Bit 12: Group Slave Absolute Phase Adjust

A low-to-high (0-1) transition of this bit creates an absolute offset between the Group's optional CAM profile and output. The offset value is taken from program variable G#_PROG_S_PH. Additional transitions of this bit will not trigger another phase adjust unless the value in G#_PROG_S_PH is modified.

The value in program variable G#_PROG_S_PH is written to G#_ABS_S_PH after the phase adjust is complete.

Bit 13: Group Local Virtual Master Absolute/Relative Mode

Absolute or relative position mode of the **Group Local** Virtual Master is set with this bit. Positioning mode of a **Group Local** Virtual Master is set with the following combination of bits **10** and **13**.

Active Position Mode	Bit 10	Bit 13
Absolute Mode	0	0
Relative Mode	0	1

Table 16-37: Setting **Group Local** Virtual Master Position Mode

Bit 14: Group Local Virtual Master Relative Trigger Mode

A low-to-high (0-1) transition triggers the relative move bit. This bit is only functional when a relative mode setup is configured using bits 10 and 13. The Group Local Virtual Master will move in increments of the value in variable G#_LM_REL_MOVE_DIST every time this bit transitions from low-to-high.

Bit 15: Group Master-Following Phase Adjust Enable

0 = OFF (default -- standard phase adjust will be performed)

1 = Group Master-Following Phase Adjust Option enabled

This bit is used to enable the Master-Following Phase Adjust Option. It is level sensitive, editable in Run Mode (phase 4), and has a default value of 0 (Off). If a phase adjust is triggered when this bit is zero (the default state), the phase adjust will be performed in the standard manner with the G#_SYNC_ACCEL and G#_SYNC_VEL values determining the phase adjustment profile and execution time. If a phase adjust is triggered when this bit is 1, the phase adjustment will be performed such that the profile and execution time are determined by the G#_MST_DIST_M_PH or G#_MST_DIST_S_PH variable and the motion of the Group's master. The value of this bit is captured when the phase adjust is triggered such that changes to the bit will not affect active phase adjustments.

Bit 16: Abort Active Master-Following Phase Adjust

0 = n/a

1 = Abort active Master-Following Phase Adjusts

This bit is used to abort active Master-Following Phase Adjusts. It is level sensitive, editable in Run Mode (phase 4), and has a default value of 0 (Off). Raising this bit during a Master-Following Phase Adjust will cause the phase adjust to stop. The Group will retain any partially executed portion of the phase adjust as an offset. The associated ELS Group Status Register phase adjustment bit can be used to confirm that the phase adjustment has been successfully aborted (the status bit will go low). This bit will not stop a standard (G#_CT_MSTR_FOL_PH = 0) phase adjust, it only affects Master-Following (G#_CT_MSTR_FOL_PH = 1) Phase Adjusts.

Register 197: Coordinated Articulation Synchronized Mode Control

A Coordinated Articulation configuration uses two control registers. This first register is used as the control register for synchronized mode. Refer to Register 198: Coordinated Articulation Local Mode Control for details on the second register. The following table lists the functions used in this control register.

Note: This register number can be assigned by Rexroth VisualMotion as a default system register when creating an icon program or can be changed to any number not reserved by Rexroth VisualMotion. It is strongly recommend that the programmer use the default number. This makes documentation and modification to user programs an easier task over the scope of the project.

Register	Register Label Name	Bit	Function
197	Coordinated Articulation Control	1	CA#_X_SYNC
		2	CA#_Y_SYNC
		3	CA#_Z_SYNC
		4	CA#_ROLL_SYNC
		5	CA#_PITCH_SYNC
		6	CA#_YAW_SYNC
		7	CA#_X_REL_MOD
		8	CA#_Y_REL_MOD
		9	CA#_Z_REL_MOD
		10	CA#_ROLL_REL_MOD
		11	CA#_PITCH_REL_MOD
		12	CA#_YAW_REL_MOD

Table 16-38: Register 197: Coordinated Articulation Control

Bits 1-6: Synchronized Mode

These bits are used to transition the Coordinated Articulated configuration axes between Sync and Local modes.

Sync Mode

A low-to-high (0-1) transition enables each axis (X, Y, Z, ROLL, PITCH and YAW) to Sync mode. This transition is delayed until the ramp generator's velocity is 0.

Local Mode

A high-to-low (1-0) transition enables each axis (X, Y, Z, ROLL, PITCH and YAW) to Local mode. Local mode is used for the manual positioning of each axis.

Bit 7-12: Positioning Mode

These bits are used to enable the absolute or relative offset calculation of the Coordinated Articulation CAM output section.

0 = Absolute Mode

1 = Relative Mode

Register 198: Coordinated Articulation Local Mode Control

A Coordinated Articulation configuration uses two control registers. This second register is used as the control register for local mode. Local mode allows the user to manually position the world coordinate in-position value to the current CAM output position before synchronizing. Refer to Register 197: Coordinated Articulation Synchronized Mode Control for details on the first register. The following table lists the functions used in this control register.

Note: This register number can be assigned by Rexroth VisualMotion as a default system register when creating an icon program or can be changed to any number not reserved by Rexroth VisualMotion. It is strongly recommend that the programmer use the default number. This makes documentation and modification to user programs an easier task over the scope of the project.

Register	Register Label Name	Bit	Function
198	Coordinated Articulation Control	1	CA#_X_MAN_EN
		2	CA#_Y_MAN_EN
		3	CA#_Z_MAN_EN
		4	CA#_ROLL_MAN_EN
		5	CA#_PITCH_MAN_EN
		6	CA#_YAW_MAN_EN
		7	CA#_X_MAN_IMD
		8	CA#_Y_MAN_IMD
		9	CA#_Z_MAN_IMD
		10	CA#_ROLL_MAN_IMD
		11	CA#_PITCH_MAN_IMD
		12	CA#_YAW_MAN_IMD

Table 16-39: Register 198: Coordinated Articulation Control

Bits 1-6: Ramp Generator Enable

These bits are used to generate a move while in Local Mode. The type of move performed (ramped or immediate) depends upon whether the axis is in Normal Local Mode or Immediate Local Mode. These bits are functional when bits 1-6 of control register 197 are set to 1 (Local Mode).

Normal Local Mode

A low-to-high (0-1) transition enables the ramp generator, creates a trapezoidal move profile for variables CA#_%_TAR_POS value to variable CA#_%_IN_POS (where # = Unit number and % = X, Y, Z, ROLL, PITCH or YAW).

A high-to-low (1-0) transition disables the ramp generator. If a move is at velocity, the velocity is ramped down using the value in variable CA#_LIN_DECEL.

Immediate Local Mode

A low-to-high (0-1) transition stores the value of variable A#_%_TAR_POS to variable CA#_%_IN_POS (essentially performing an immediate move).

A high-to-low (1-0) transition has no effect.

Bits 7-12: Immediate Mode

These bits are used to set a Coordinated Articulated axes' local mode positioning to Normal or Immediate.

0 = Normal Local Mode

1 = Immediate Local Mode

Registers 241 and 242: Virtual Master 1 & 2 Status

Note: These register numbers can be assigned by Rexroth VisualMotion as default system registers when creating an icon program or can be changed to any number not reserved by Rexroth VisualMotion. It is strongly recommend that the programmer use the default numbers. This makes documentation and modification to user programs an easier task over the scope of the project.

Register	Register Label Name	Bit	Function
241 and 242	Virtual Master 1 and 2 Status	1	VM#_ST_FSTOP
		2	VM#_ST_HOME
		3	Reserve3
		4	VM#_ST_VMODE
		5	VM#_ST_RELMODE
		6	Reserve6
		7	VM#_ST_ZEROVEL
		8	VM#_ST_INPOS
		9-16	Not used

Table 16-40: Registers 241-242: Virtual Master 1 & 2 Status

Bit 1: Virtual Master 1or 2 Fast Stop Active

An active high (1) is an indication that the fast stop function for this Virtual Master is active in bit 1 of the Virtual Master control register.

Bit 2: Virtual Master Home Position

An active high (1) is an indication that the homing process for this Virtual Master is complete and at the command position in variable VM#_HOME_POS.

Bit 4: Virtual Master Mode of Operation Active

This status bit indicates which of the following modes of operation are active.

0 = Positioning Mode

1 = Velocity Mode

Bit 5: Virtual Master Absolute/Relative Mode Active

This status bit indicates whether absolute or relative position mode of the Virtual Master is active.

0 = Absolute Mode

1 = Relative Mode

Bit 7: Virtual Master at Zero Velocity

This status bit is active high (1) anytime the Virtual Master is at zero velocity.

Bit 8: Virtual Master in Position

This status bit is active high (1) when the Virtual Master has reached its commanded position. This can occur when…

- Virtual Master stop position variable VM#_STOP_POS is reached.
- Virtual Master absolute position variable VM#_CMD_ABS_POS is reached.
- Virtual Master relative distance variable VM#_REL_MOVE_DIST is reached.

Registers 243 - 250: ELS Groups 1-8 Status

Note: These register numbers can be assigned by Rexroth VisualMotion as default system registers when creating an icon program or can be changed to any number not reserved by Rexroth VisualMotion. It is strongly recommend that the programmer use the default numbers. This makes documentation and modification to user programs an easier task over the scope of the project.

Register	Register Label Name	Bit	Function
243-250	ELS Group 1-8 Status	1	G#_ST_LOCK_ON
		2	G#_ST_M_REL_PH
		3	G#_ST_S_REL_PH
		4	G#_ST_MSTR_SEL
		5	G#_ST_VAR_ACK
		6	G#_ST_LOCAL
		7	G#_ST_LM_FSTOP
		8	G#_ST_LM_HOME
		9	G#_ST_MOTION
		10	G#_ST_LM_VMODE
		11	G#_ST_M_ABS_PH
		12	G#_ST_S_ABS_PH
		13	G#_ST_LM_RELMODE
		14	G#_ST_LM_ZEROVEL
		15	G#_ST_LM_INPOS
		16	Not Used

Table 16-41: Registers 243-250: ELS Group 1-8 Status

Bit 1: Group Lock Off/ Lock On Cams Status

An active high (1) is an indication that the lock off cam profile feature in ELS is active. This cam profile decelerates to a stop over one cycle of the master.

A low (0) is an indication that the lock on cam feature is active. This profile accelerates from a stopped position to match the master's velocity over one cycle of the master (360 degrees). Once the velocity is matched, a one-to-one cam profile is active and follows the master.

Bit 2: Group Master Phase Adjust Status

A low-to-high (0-1) transition of the ELS Group control register bit 2 (G#_CT_M_REL_PH) cause a momentary transition (0-1-0) of this group master phase adjust status bit.

Bit 3: Group Slave Phase Adjust Status

A low-to-high (0-1) transition of the ELS Group control register bit 3 (G#_CT_S_REL_PH) cause a momentary transition (0-1-0) of this group master slave adjust status bit.

Bit 4: Group Master Input Select Status

A low-to-high (0-1) transition of the ELS Group control register bit 4 (G#_CT_MSTR_SEL) cause an active high of this status bit.

A 0 is an indication that the ELS Group and slave axes are following the master 1 input to the group as selected in the user program and in the integer block variable (G#_MSTR1).

A 1 is an indication that the ELS Group and slave axes are following the master 2 input to the group as selected in the user program and in integer block variable (G#_MSTR2).

Bit 5: Group Forcing Status

This status bit is active high (1) when the ELS group is switched to local mode and the group forcing bit (G#_CT_VAR_CLK) is active. This status bit is an indication that one of the following internal variables is being forced to a different value.

- internal group input master position
- group cam table input position
- ELS Group master position (only when bit 9 in ELS group configuration word is set to 1)
- state of the state machine for lock on / lock off
- absolute group master and group slave offset
- lock on, lock off and 1:1 cam profile ID number
- H-factor for lock on, lock off and 1:1 cam profile
- M and N factor

Bit 6: Group Local Mode Select

This status bit is active high (1) when the ELS Group is switched to local mode and the master input velocity is at zero. Once active high, the group's internal variables can be forced or jogged.

A high-to-low (1-0) transition of the group's control register bit 6 (G#_CT_LOCAL) switches the group back to the selected group input master and cause a high-to-low transition of this status bit.

Bit 7: Group Local Virtual Master Fast Stop Active

An active high (1) is an indication that the fast stop function for the Group Local Virtual Master is active in bit 7 of the Group control register.

Bit 8: Group Local Virtual Master Home Position

An active high (1) is an indication that the homing process for the Group Local Virtual Master is complete and at the command position in variable G#_LM_HOME_POS.

Bit 9: Group Motion Status

This bit is active high (1) whenever the group is in motion, whether following a master input or being jogged.

Bit 10: Group Local Virtual Master Mode of Operation Active

This status bit indicates which of the following modes of operation are active.

0 = Positioning Mode

1 = Velocity Mode

Bit 11: Group Master Absolute Phase Adjust Status

A low-to-high (0-1) transition of the ELS Group control register bit 11 (G#_CT_MSTR_PH_ABS) cause a momentary transition (0-1-0) of this group master phase adjust status bit.

Bit 12: Group Slave Absolute Phase Adjust Status

A low-to-high (0-1) transition of the ELS Group control register bit 12 (G#_CT_SLV_PH_ABS) cause a momentary transition (0-1-0) of this group slave phase adjust status bit.

Bit 13: Group Local Virtual Master Absolute/Relative Mode Active

This status bit indicates whether absolute or relative position mode of the **Group Local Virtual Master** is active.

0 = Absolute Mode

1 = Relative Mode

Bit 14: Group Local Virtual Master at Zero Velocity

This status bit is active high (1) anytime the **Group Local Virtual Master** is at zero velocity.

Bit 15: Group Local Virtual Master in Position

This status bit is active high (1) when the **Group Local Virtual Master** has reached its commanded position. This can occur when…

- Group Local Virtual Master stop position variable G#_LM_STOP_POS is reached.
- Group Local Virtual Master absolute position variable G#_LM_CMD_ABS_POS is reached.
- Group Local Virtual Master relative distance variable G#_LM_REL_MOVE_DIST is reached.

Register 288: Coordinated Articulation Synchronized Mode Status

A Coordinated Articulation configuration uses two status registers. This first register is used to monitor the status of each axis enabled to synchronized mode. It also monitors the equivalence between variables CA#_%_IN_POS and CA#_%_TAR_POS (where # = Unit number and % = X, Y, Z, ROLL, PITCH or YAW). Refer to Register 289: Coordinated Articulation Local Mode Status for details on the second register. The following table lists the functions used in this status register.

Note: This register number can be assigned by Rexroth VisualMotion as a default system register when creating an icon program or can be changed to any number not reserved by Rexroth VisualMotion. It is strongly recommend that the programmer use the default number. This makes documentation and modification to user programs an easier task over the scope of the project.

Register	Register Label Name	Bit	Function
288	Coordinated Articulation Status	1	CA#_X_SYNCED
		2	CA#_Y_SYNCED
		3	CA#_Z_SYNCED
		4	CA#_ROLL_SYNCED
		5	CA#_PITCH_SYNCED
		6	CA#_YAW_SYNCED
		7	CA#_X_READY
		8	CA#_Y_READY
		9	CA#_Z_READY
		10	CA#_ROLL_READY
		11	CA#_PITCH_READY
		12	CA#_YAW_READY

Table 16-42: Register 288: Coordinated Articulation Status

Bits 1-6: Synchronized Mode

These bits are used to indicate if a Coordinated Articulation axis is enabled to Sync or Local modes.

0 = Local Mode (using target position for world coordinated input value)

1 = Sync Mode (using CAM output world coordinated input value)

Bit 7-12: Positioning Mode

These bits are used to indicate if a Coordinated Articulation axis world coordinated input value equals the CAM output value.

0 = Not ready for synchronization

1 = CAM output value (CA#_%_CAM_OUT) equals world coordinated input value (CA#_%_IN_POS) and that the ramp generator is at 0 velocity.

Register 289: Coordinated Articulation Local Mode Status

A Coordinated Articulation configuration uses two status registers. This second register is used to monitor if a Coordinated Articulation axis is at position when moving the axis in local mode. Refer to Register 288: Coordinated Articulation Synchronized Mode Status for details on the first register. The following table lists the functions used in this status register.

Note: This register number can be assigned by Rexroth VisualMotion as a default system register when creating an icon program or can be changed to any number not reserved by Rexroth VisualMotion. It is strongly recommend that the programmer use the default number. This makes documentation and modification to user programs an easier task over the scope of the project.

Register	Register Label Name	Bit	Function
289	Coordinated Articulation Status	1	CA#_X_AT_TAR_POS
		2	CA#_Y_AT_TAR_POS
		3	CA#_Z_AT_TAR_POS
		4	CA#_ROLL_AT_TAR_POS
		5	CA#_PITCH_AT_TAR_POS
		6	CA#_YAW_AT_TAR_POS

Table 16-43: Register 289: Coordinated Articulation Status

Bits 1-6: Local Mode At Target Position

These bits are set to 1 when a Coordinated Articulated axis target position (CA#_%_TAR_POS) variable is equal to its respective in position (CA#_%_IN_POS) variable.

0 = Refer to Note

1 = CA#_%_IN_POS at CA#_%_TAR_POS position and ramp generator is at 0 velocity.

Note: This bit is not set to 1 if the ramp generator move is interrupted (dropped Enable, user task stopped, etc.) before the target position is reached. In such an instance, it is not possible to 'force' the bit to 1 by setting the ramp generator's target value equal to the current input value. Rather, another ramp generator move must be completed before this bit is set to 1.

17 Debugging and Monitoring

17.1 Finding Program Problems

Identifying errors in complex programs can be difficult. A thorough understanding of your application and a project designed around the application will help minimize the difficulty in locating errors. A practical approach to designing a project is to begin with simple, basic program blocks. Test the blocks, independently if possible, even if testing requires writing additional program just for test purposes. A tested and dependable section of a program allows you to focus on just the potential problem areas. If the program compiles correctly, make sure that the problem lies with the program, not the hardware. If necessary, write short test programs to test individual hardware functions.

Use a program branch and the VisualMotion's message capabilities to insert a message into your program. Shortly stopping the program and checking critical values can tell you where errors are occurring.

Think through the implications of using triggered events. Remember that events and the execution of event functions typically occur asynchronously to program tasks. You cannot always depend on the timing of triggered events. It may be necessary to add additional program code to provide synchronization.

The following Task parameters can also be used to help with program debugging:

- T-0-0130 Current Instruction Pointer
- T-0-0131 Current Instruction
- T-0-0132 Instruction Pointer at Error
- T-0-0133 Composite Instruction Pointer
- T-0-0135 Current Subroutine
- T-0-0136 Stack Variable Data
- T-0-0137 Task Subroutine Breakpoint
- T-0-0138 Sequencer Information
- T-0-0200 Last Active Event Number

17.2 Icon Language Warnings and Error Messages

VisualMotion Icon Compiler generates the following warning messages. After receiving a warning message you may continue or exit the compilation.

- Data missing in one or more fields, do you still wish to continue?
- Caution! Changing Modes may halt motion. Continue?
- Caution! Changing Modes may start motion. Continue?
- File does not contain source program!
- Icon workspace at end is not empty. Program parts will be lost, continue anyway?
- Transfer failed!

VisualMotion's Icon Compiler displays the following error messages:

- More than one connect icon with number %d!

- Function variables must be defined first!
- Data Size icon objects exceed size of non-volatile ram!
- Change to default registers and variables for this number?
- Only one ELS System Master icon allowed per program!
- Only one Virtual Master icon allowed per program!
- Only eight ELS Groups allowed!
- Only ten PIDs allowed!
- Only four CAM Indexers allowed!
- Warning! Frequent changes of static drive parameters can cause premature failure of it's non-volatile memory.
- Valid Entries are '0' or '1'.
- Invalid name!
- Cannot change task or open dialog window while dialog window is open
- Axis undefined or not unique.
- Valid event numbers are 1 to 100.
- Valid axis numbers are 1 to 64.
- Valid number range is 1 - 32767.
- Valid percents are 1 - 100.
- Labels must start with an alpha character!
- Label name already exists!
- Number missing or out of range.
- Selected Icon is not a subroutine or no icon selected!
- Data Field Empty!
- Label type must be defined!
- Task name undefined.
- No filename specified.
- Non-Branch icons have only two output connections.
- Branch icons have only two output connections.
- Point out of range.
- Connection could not be made, try connecting adjacent ---?
- Only connections between icons or adjacent blocks can be ---?
- Finish icon not found or open path!
- Start icon not found or multiple Start icons found!
- Icon program not found!
- Cannot open code file!
- Unknown icon term _______.
- Missing axis selection.
- Open in program flow, at or near highlighted icon, ---?
- Branch Icon has missing connection or one in wrong dire ---?
- No axis selected!
- Time Delay out of range!
- Could not initialize update timer!
- Operation type not selected!
- Drive numbers doesn't match.

- Should drive number be c ---?
- Can't open file ________!
- Source or target not selected!
- Valid range ___ - ___
- Valid range ___ - ___
- CONTROL card parameters cannot be changed!
- File syntax other than parameters!
- File of different type parameters!
- CONTROL card is not communicating!
- No selection made!

Second Pass Compiler Errors

Line xx, more than one equal operator.

On line xx, more than one "=" character was found.

Line xx, colon used for other than mark!

A colon was found beyond the first word on line xx.

Line xx, function start found inside of subroutine!

A Start icon was found inside a subroutine on line xx.

Line xx, function end found without function start!

A Finish icon was found without first finding a Start icon on line xx.

Duplicate local argument 'xx' found in subroutine 'yy'!

Two local arguments with the same name xx were found in subroutine yy.

Subroutine 'xx' has more than 5 user accessible arguments!

A subroutine can only have 5 arguments passed to it. Subroutine xx has more than 5.

Subroutine 'xx' has more than 16 local variables/arguments!

A subroutine can only have 16 local or stack variables. Subroutine xx has more than 16.

Line xx, invalid sequencer list index 'xx'!

An error was made while defining a sequencer on base code line xx. One of the sequencer "list_Numbers" is greater than 30 or has been entered out of sequence (0,1,2,3,4,5).

Line xx, invalid sequencer step index 'yy'!

An error was made while defining a sequencer step on base code line xx. One of the sequencer steps "step_Numbers", yy, is greater than sequencer functions defined in the "DATA/SIZE instruction of the program or has been entered out of sequence (0, 1,2,3,4,5).

Number of sequencer step names exceed sequencer step size!

The number of sequencer steps found is greater than sequencer Steps defined in the "DATA/SIZE instruction of the program.

Number of sequencer names exceed sequencer list size!

The number of sequencer names found is greater than sequencer Lists defined in the “DATA/SIZE instruction of the program.

Line xx, invalid axis number - yy!

An error was found in the PLS/INIT instruction on base code line xx. The “axis” number yy is not valid for the type selected. Valid ranges are:

Type	Range
1or 2	1-2
3 or 4	1-64
5	1-6
6	1-8

Line xx, invalid PLS master type - yy, range 1 – 4.

An error was found in the PLS/INIT instruction on base code line xx. The “type” yy is not a value from 1 to 4.

Line xx, duplicate label ‘yy' or multiple definition of variable!

An error was found in the FUNCTION/ARG instruction or START icon on base code line xx. The label yy was used already.

Line xx, error in number of function arguments - yy!

An error was found in the CALL instruction or SUB icon on base code line xx. The number of arguments yy, passed to the subroutine is different than defined in the function called.

Line xx, index 'yy' is a float!

An error was found on base code line xx. The index used for a variable is a float (i.e. F[F5]).

Line xx, two names assigned to a sequencer 'yy'!

An error was made while defining a sequencer on base code line xx. A sequencer index ‘yy’ is given two different names.

Line xx, same name assigned to two sequencers 'yy'!

An error was made while defining a sequencer on base code line xx. The same name ‘yy’ is given to two sequencers.

Line xx, invalid cam option type 'yy'!

An error was made while defining the CAM/BUILD instruction on base code line xx. The cam option or type ‘yy’ is outside the range 1-4.

Line xx, end point 'yy' is less than start point!

An error was made while defining the CAM/BUILD instruction on base code line xx. The point defined as the end_point ‘yy’ is less than the start point.

Line xx, invalid cam wait option - yy, range 0 - 1

An error was made while defining the CAM/BUILD instruction on base code line xx. The wait option ‘yy’ is outside the range 0-1.

Maximum number of messages reached!

The number of messages, status and diagnostic, allowed per program is 500. An attempt to exceed this was found.

Multiple PLS initializations found!

More than one instruction was found to define the same PLS data .

Line xx, invalid message type, range 1 - 2

An error was made in the MESSAGE instruction on base code line xx. The valid range of values are 1-2.

Line xx, invalid cam type 'yy'! 0=CLC, 1=Drive.

An error was made in the CAM/ENGAGE instruction on base code line xx. The value ‘yy’ is invalid, valid range of values is 0-1.

Line xx, ELS slave 'yy' same as master!

An error was made in the ELS/INIT instruction on base code line xx. The slave axis ‘yy’ is the same as the master axis.

Line xx, invalid PID number 'yy', range 1 - 10

An error was made in the PID/CONFIGURE instruction on base code line xx. The loop number ‘yy’ is invalid, range is 1-10.

Line xx, invalid PID type 'yy', range 1 - 1

An error was made in the PID/CONFIGURE instruction on base code line xx. The type ‘yy’ is invalid, the only type available is 1.

Line xx, same PID status and control registers 'yy'

An error was made in the PID/CONFIGURE instruction on base code line xx. The same register number ‘yy’ was used for the control and status, they must be different.

Line xx, invalid PID loop time 'yy', range 8 - 152

Line xx, PID loop time 'yy', not multiple of 8

An error was made in the PID/CONFIGURE instruction on base code line xx. Loop times are multiples of 8 ms, from 8 to 152. The value ‘yy’ is not valid.

Line xx, Data initialization 'yy', exceeds data range

Line xx, variable block 'yy' exceeds variable allocation!

An error was made in the VAR/INIT instruction on base code line xx. An attempt was made to initial variables beyond their range with ‘yy’. Increase size of variables in DATA/SIZE instruction.

Line xx, Multiple configurations For PID loop yy

An error was made in the PID/CONFIGURE instruction on base code line xx. More than one initialization found for PID ‘yy’.

Line xx, PID control blocks overlapping ‘yy’

An error was made in the PID/CONFIGURE instruction on base code line xx. A float variables control block overlaps another.

Line xx, invalid PID argument ‘yy’

An error was made in the PID/CONFIGURE instruction on base code line xx. Invalid set_point_type, feedback_type, or output_type found ‘yy’, valid range 1-4.

Line xx, zone element 'yy' missing or entered with spaces!

Line xx, zone element 'yy' unknown!

An error was defining a zone element instruction on base code line xx. Invalid text found was ‘yy’.

Line xx, Missing open parenthesis!

An error was found in a mathematical expression on base code line xx. A closed parenthesis found without matching open.

Line xx, invalid ELS Group number 'yy', range 1 – 8

An error was made in the ELS_GROUP instruction on base code line xx. Invalid group number found ‘yy’, valid range 1-8.

Line xx, multiple ELS Master instructions found!

A second ELS_MASTER instruction was found on base code line xx. Only one ELS_MASTER instruction is allowed per program.

Line xx, multiple ELS Group ‘yy’ instructions found!

A second ELS_GROUP instruction for group ‘yy’ was found on base code line xx. Only one ELS_GROUP instruction per group is allowed in a program.

Line xx, axis 'yy' found in multiple ELS Group instructions!

A second ELS_GROUP instruction for axis ‘yy’ was found on base code line xx. An axis can only be assigned to one ELS_GROUP.

Line xx, invalid ELS Master number 'yy', range 1 - 6

An error was made in the ELS_MASTER instruction on base code line xx. Invalid master number found ‘yy’, valid range 1-6.

Line xx, Valid modes are 0=axis, 1=ELS Master, 2=ELS Group !

An error was made in the ROTARY/EVENT instruction on base code line xx. Valid modes are 0=axis, 1=ELS Master, 2=ELS Group

Line xx, invalid Virtual Master number 'yy', range 1 - 2

An error was made in the V_MASTER instruction on base code line xx. Master number ‘yy’ is not in the range 1-2.

Line xx, Illegal syntax : syntax 'yy' is not allow at the moment.

An error was made in the mathematical equation instruction or Calc icon on base code line xx. Syntax ‘yy’ is not allowed in this sequence of terms.

'*xxxx*' - unresolved mark reference.

The mark '*xxxx*' was used as a destination in a branch or subroutine call, but was not found in the code. Check for possible spelling error or missing subroutine.

Line *xx*, all probe types zero or not unique!

The probe arguments are both zero or are the same.

Line *xx*, argument '*yyyy*' out of range!

The argument *'yyyy'* is out of range, check syntax in manual.

Line *xx*, axes missing or not unique!

In a AXES_GROUP command for ratioed axis, the slave axis argument is zero or is the same as the master axis.

Line *xx*, axis number '*yyyy*' out of range (*wwww*, *xxxx, 1-zzzz*).

The axis number or label '*yyyy*' has not been resolved to a valid number. The numbers '*wwww*', '*xxxx*', and range 1 to '*zzzz*' are valid axis numbers.

Line *xx*, bit number '*yyyy*' out of range (1-16)!

On line '*xx*', the string '*yyyy*' is evaluated to number outside of the valid range for register bits.

Line *xx*, 'compare' arguments must be floats, integers, or constants!

Compare arguments must be Fx, GFx, GIx, Ix or equivalent labels or constants. Compares are derived from "BRANCH" icons in user programs.

Line *xx*, event element '*yyyy*' missing or entered with spaces!

On line 'xx', the compiler has not found a "]" in the event string 'yyyy'. It uses this to position to the start of the event element. The event element { s, t, d, a, f, m }must follow immediately.

Line *xx*, event element '*yyyy*' unknown!

The event element '*yyyy'* was not found in the event element table, check manual for exact syntax.

Line *xx*, event EVT[].*yy* data is not changeable in program

Line *xx*, event function '*yyyy*' not found in program!

The event function 'yyyy' was not found in the program. Check spelling and capitalization.

Line *xx*, event message '*yyyy*' must start with quotes!

The compiler is expecting a quote to start the ASCII string for the event message, but did not find it.

Line *xx*, event number '*yyyy*' out of range!

On line '*xx*', the string '*yyyy*' was evaluated to be out of the range for events defined for this program. Events and other variables are declared in the VM Data Table.

Line *xx*, float number '*yyyy*' conversion error!

The string 'yyyy' for conversion to a float was determined to contain one of the following errors:

No numeric characters.

More than one exponent symbol ‘E’ ('e').

More than two sign symbols'.

More than one decimal point.

Alpha characters other than ‘E’ ('e').

Line *xx*, hex number '*yyyy*' conversion error!

On line '*xx*', the string '*yyyy*' is greater than 10 characters long or contains non-hexadecimal characters. Valid strings start with 0x and contain ASCII characters 0-9, A-F or a-f (0x1BF8).

Line *xx*, integer number '*yyyy*' conversion error!

The string 'yyyy' for conversion to an integer was determined to contain one of the following errors:

No numeric characters.

Number of numeric characters exceed 10.

The converted number exceeds 0x7FFFFFFF.

Line *xx*, Invalid argument '*yyyy*'!

Line *xx*, Invalid cam number '*yyyy*'! Range 1 to 8.

The CAM number *'yyyy'* was evaluated to be less than one or greater than 8.

Line *xx*, Invalid count or count plus register exceeds range!

The count of registers to be transferred was evaluated to be less than one or when added to the starting register exceeds the maximum register range (512 registers for GPP).

Line *xx*, Invalid Encoder type '*yyyy*', 1=primary, 2=secondary!

The ELS master encoder type *'yyyy'* was evaluated to be less than one or greater than 2.

Line *xx*, Invalid ELS type '*yyyy*', range 1 to 4!

The ELS type *'yyyy'* was evaluated to be less than one or greater than 4.

Line *xx*, Invalid sync type '*yyyy*', 1=velocity, 2=phase, 3=cam!

The ELS sync type *'yyyy'* was evaluated to be less than one or greater than 3.

Line *xx*, Left term '*yyyy*' of equation must not be constant!

A calculation must have a variable(Fx, GFx, GIx, Ix) or changeable table element(ABS[1].x, EVT[3].d, etc.) as its term to the left of the equal sign.

Line *xx*, Maximum number of terms reached.

When parsing the line 'xx', the number of terms exceeded 32. A term is one or more alphanumeric characters followed by a space, comma or other non-alphanumeric character. This error usually only occurs in message statements with many short words. Try a message with fewer words.

Line *xx*, Maximum size (20) of term exceeded!

While parsing line *'xx'* for arguments a string of more than 20 characters was encountered. Arguments and argument labels are limited to 20 characters. Check label length and use of commas between arguments.

Line *xx*, Message exceeds 80 characters!

The number of characters used in the message exceeds 80 characters. This count includes spaces.

Line *xx*, missing argument(s)!

One or more additional arguments were expected.

Line *xx*, missing beginning quotes of message!

On line 'xx', quotes were expected to denote the start of the message. Also, use quotes when using the "CALC" icon to set an event message.

Line *xx*, missing closing bracket ']'!

The closing bracket used to denote the end of the index of a data structure was not found.

Line *xx*, missing closing curly brace '}'!

The closing brace used to denote the end of initialization data for a data structure was not found. Other causes are extra arguments or the wrong character.

Line *xx*, missing closing quotes of message!

On line 'xx', quotes were expected to denote the end of the message. Also, use quotes when using the "CALC" icon to set an event message.

Line *xx*, missing mark name!

The argument of branch command does not start with an alpha character. Check for missing or misspelled argument.

Line *xx*, Parameter <type> must be 'A', 'C', 'D' or 'T'

The parameter class was not found to be 'A', 'C', 'D', 'T', or equivalent label. Check for missing or misspelled argument.

Line *xx*, point element '*yyyy*' missing or entered with spaces!

On line 'xx', the compiler has not found a "]." in the point string 'yyyy'. It uses this to position to the start of the point element. The point element {x, y, z, b, s, a, d, j, e1, e2, e3, e4, r, p, ya, el} must follow immediately.

Line *xx*, point element '*yyyy*' unknown!

The point element '*yyyy'* was not found in the point element table, check manual for exact syntax.

Line *xx*, register number '*yyyy*' out of range (1-*zzzz*)!

The register number '*yyyy*' is less than one or greater than the maximum register '*zzzz*'.

Line *xx*, table or array index out of range '*yyyy*'!

The table or array index '*yyyy*' is less than one or greater than the number declared by DATA/SIZE command or by the default declaration.

Line *xx*, table or array label index out of range '*yyyy*'!

The table or array index label '*yyyy*' is evaluated to be less than one or greater than number declared by DATA/SIZE command or by the default declaration.

Line *xx*, Task must be 'A', 'B', 'C' or 'D'!

The compiler is expecting a task argument (A, B, C, or D) and has not found it. This may result from a missing argument or arguments out of sequence.

Line *xx*, too many arguments!

More terms than expected were found following the command. Check for extra arguments, extra commas or terms with spaces in them.

Line *xx*, unknown mnemonic operator - '*yyyy*'!

On line '*xx*', the string '*yyyy*' is assumed to be a command, but was not found in the list of valid commands.

Line *xx*, unknown or out of range variable '*yyyy*'!

On line '*xx*', the string '*yyyy*' is not of the type expected. Check for argument type(float where integer should be used, etc.) , or for missing or misspelled arguments.

Line *xx*, unresolved index '*yyyy*'!

The index '*yyyy*' could not be resolved, check for missing or misspelled label. Labels are case sensitive and cannot contain spaces.

Line *xx*, unresolved index label '*yyyy*'!

The index label '*yyyy*' could not be resolved to an integer or integer variable, check for missing or misspelled labels. Labels are case sensitive and cannot contain spaces.

Line *xx*, Unsupported structure transfer!

The data structures equated to each other are not of the same type. The data structure transfers supported are: Point to Point, Event to Event, and Zone to Zone.

Line *xx*, Valid modes are 1=single axis, 2=ELS synchronized!

The second argument of the "ELS/MODE" command is missing or out of range. This can also be generated if the first argument is invalid and appears as two or more arguments to the compiler.

Line *xx*, variable table '*yyyy*' index unknown!

The closing bracket is missing or other delimiters found in the index term of a variable or register with index format.

Mark table filled - *yyyy*, reduce number of subroutine calls.

The total number of marks used exceeds the table space provided. Marks are the location tags of the start of tasks, event functions and subroutines, or, the destination of a branch or Goto. Try to optimize your program to reduce the number of branches. If the problem persists, contact your Rexroth Indramat representative.

Upon successful completion of the compile, the number of marks and labels used is displayed in the completion window.

No main task (A, B, C, or D) found!

After compiling the program, no marks were found for Task_A, Task_B, Task_C, or Task_D. One or more of these task marks must be used. The marks for the main tasks are not case sensitive.

Sequencing error in output file!

While computing byte offsets for branches and subroutine calls, an unknown command op-code was encountered. This error can occur in a corrupted Windows memory system or a compiler bug. Try rebooting your computer and compiling again. If the problem persists, contact your Rexroth Indramat representative.

Size of program exceeds compiler space!

The compiler has 48k of space available for program development, this error occurs when that space is filled. Variables and tables are not included in this space. Try reworking your program to fit it in the space.

Unable to allocate memory for compiler!

The 2nd pass compiler uses a large block of memory (48K) allocated from the Windows operating system to build the program. When Windows fails to allocate this memory, this error occurs. Try closing other applications or rebooting Windows to free needed memory.

Unable to open source file.

This error is issued on failing to open the file "CLCCODE.TXT." Some possible causes are:

File "CLCCODE.TXT" is not in the "\\Indramat\VisualMotion 8\" directory. This file is created by compiling an icon program.

The maximum number of files is already open. DOS file "CONFIG.SYS" configures the maximum number of files.

The file is already open and cannot be shared.

Write to file error!

This error occurs when the number of bytes sent to the output file doesn't match the number of bytes written in the output file. Check for available hard drive disk space or write protection on the output file.

18 Communication Servers

18.1 Overview

Rexroth VisualMotion 11 supports the following communication servers:

Communication Server	Protocol	Target Firmware	VisualMotion Toolkit Support	IndraLogic Support
SCP (Scalable Communication Platform)	SIS	GPP 9, 10, or 11 GMP 9, 10, or 11	Yes	Yes
VM Dde (VisualMotion Dynamic Data Exchange)	ASCII	GPP 7, 8, 9, 10, or 11 GMP 9, 10, or 11	Yes	No

Table 18-1: Communication Servers

The basic features of each server are listed in the following table:

Feature	Scalable Communication Platform (SCP)	VisualMotion Dde Server (VM Dde)
Dde Interface	Yes	Yes
OPC Interface	Yes	No
COM Interface	Yes	No
Serial Communication	Yes	Yes
Ethernet Communication	Yes	Yes
PCI-Bus Communication	Yes (PPC-P11.1)	No

Table 18-2: SCP and VM Dde Features

SCP Server

The SCP communication server uses Dde, OPC, and COM interface servers to communicate with different client softwares.

Interface Server	Description
Dde	communicates with Windows based softwares, such as VisualMotion Toolkit
OPC	communicates with Windows based HMI client softwares supporting the VT variant command structure.
COM	communicates with PLC softwares, such as IndraLogic

Table 18-3: SCP Interface Servers

The SCP communication server converts the multiple forms of communication into a universal SIS protocol. The SIS protocol is then used to communicate with VisualMotion controls using an Ethernet, Serial, or PCI-Bus(PPC-P11.1) driver. Refer to the *chapter 19, SIS Communication,* for more information.

Note: The Wonderware HMI, OPCLink in INTouch version 7, does not work with the VisualMotion 11 OPC server. Refer to the Wonderware website for information on upgrading to OPCLink version 7.6 to use the OPC server.

VM Dde Server

The VisualMotion Dde server uses ASCII protocol to communicate with VisualMotion Toolkit (motion-only). Although the server is available for GPP or GMP 10 projects, it is available with VisualMotion 11 in support of pre-GPP 10 firmwares.

The VisualMotion Dde server does not support the SIS protocol and can not be used to communication with IndraLogic. If the VisualMotion Dde server is used for a VisualMotion 11 project, the only communication options available for IndraLogic are Direct RS232 Connection via X16 and Direct EtherNet Connection. Refer to *IndraLogic Communication Channels* in section 7.1 of volume one for details.

Refer to the *VisualMotion 9 Application* manual for configuring the VM Dde server.

18.2 SCP Server Application Files

SCP and applications to support SCP and OPC communication are automatically installed with VisualMotion 11 software, including:

Application	Description
DDESCP.exe	launches the Dde communication server
OPCClient.exe	launches the OPC Client interface
OPCScp.exe	launches the OPC communication server
ScpServer.exe	launches the SCP COM server
ScpSyscon.exe	launches the SCP Systemconfigurator
ScpTrace.exe	launches the SCP TraceMonitor

Table 18-4: SCP Server Applications

These application files are installed in the following default folder location:

C:\Program Files\Common Files\Rexroth\SCP\Bin.

They can also be found from the Windows Start Menu under ***Start*** ⇒ ***Programs*** ⇒ ***Rexroth*** ⇒ ***SCP***.

18.3 Configuring the SCP Server

Use the following procedure to configure and establish SIS protocol communication between VisualMotion Toolkit and a VisualMotion control:

1. Open VisualMotion Toolkit and select the ***View and edit control data in "Service" mode*** radio button.
2. Select ***Tools ⇒ Control Selection***.
3. From the *Control Selection* window, select the SCP (SIS Protocol) radio button.

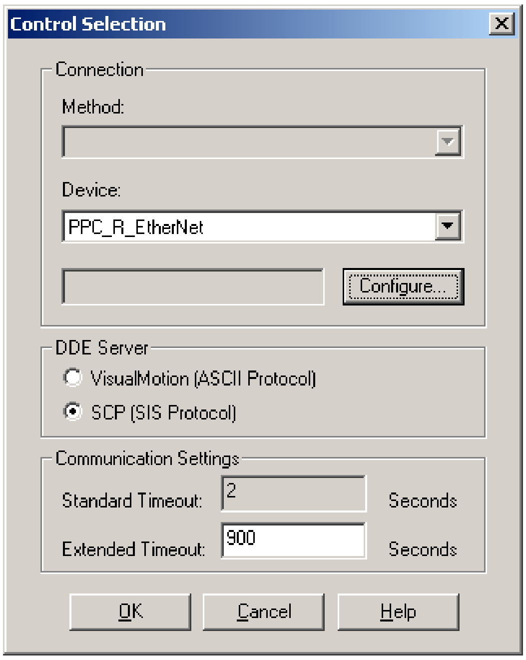


Fig. 18-1: Control Selection

4. Click the **Configure…** button to open the SCP Systemconfigurator shown in the following figure.

Note: The SCP Systemconfigurator can also be launch without using VisualMotion Toolkit by executing the ScpSyscon.exe file.

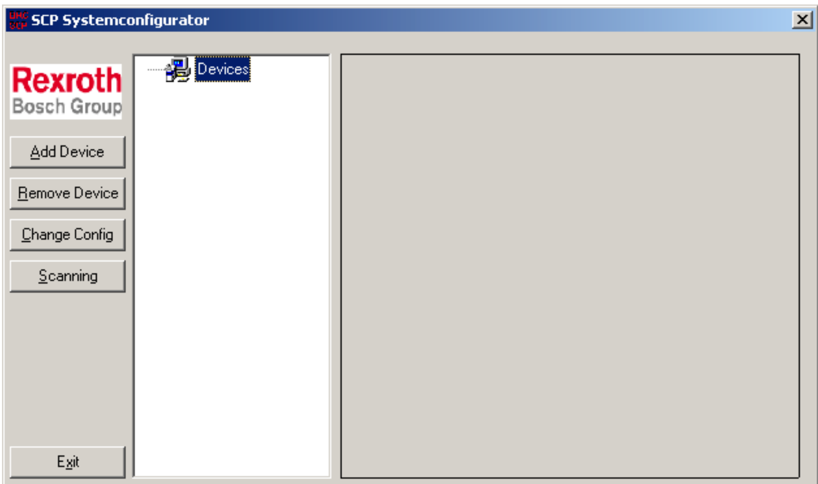
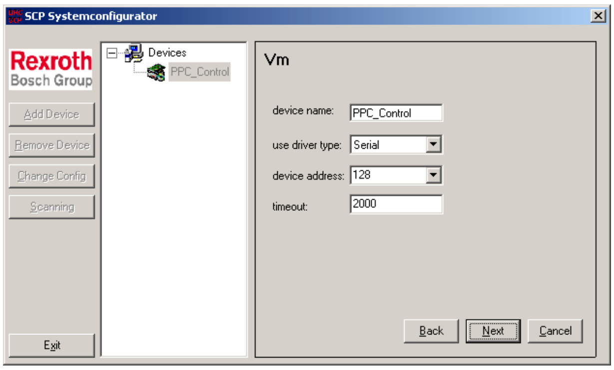


Fig. 18-2: SCP Systemconfigurator

5. In the *SCP Systemconfigurator* window, click ***Add Device***.
6. Select VM (VisualMotion) as the device control and click ***Next***.
7. Configure the device by entering a name, driver type (Ethernet, PCI, or Serial), device address, and timeout.

Note: The device address should match the address listed in control parameter C-0-0002. Scroll through the H1 display by pressing the S1 button on the control until the Unit number is displayed. Device address 128 is a point-to-point connection that generates a response from the control regardless of the setting in C-0-0002.

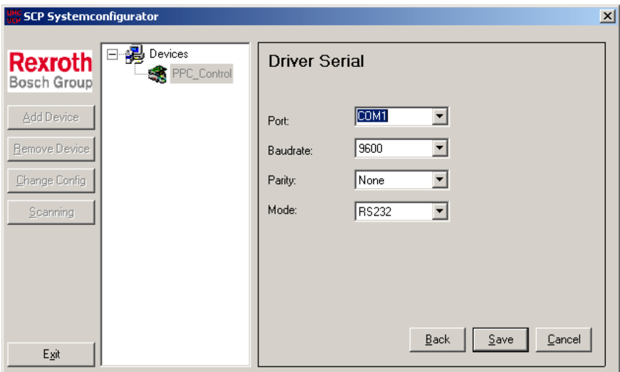


Syscon_Device.tif

Fig. 18-3: SCP Systemconfigurator Window Device Configuration

8. Click ***Next*** to configure the selected driver type.
9. Refer to step 10 for configuring a Serial driver, step 12 for configuring an EtherNet driver, or Step 13 for configuring a PCI driver.

Configuring a Serial Driver



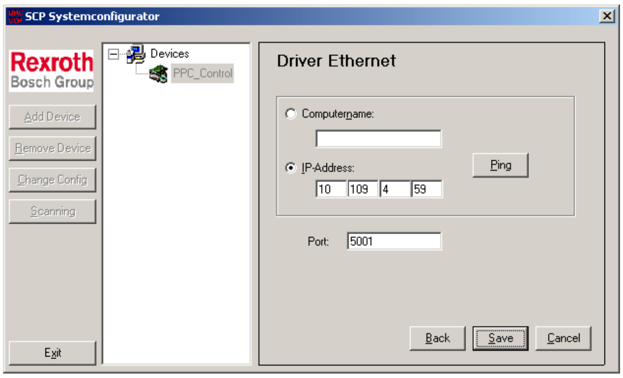
Syscon_Serial_Driver.tif

Fig. 18-4: Serial Driver Settings

10. Select the Port that matches the connection from the PC to the control.
11. Select the Baud rate, set the Parity to None, choose the interface Mode and click ***Save***.

Note: Scroll through the H1 display by pressing the S1 button on the control to display the current baud rate and mode settings for serial connection X10 or X16.

Configuring an EtherNet Driver



Syscon_Driver_Ethernet.tif

Fig. 18-5: SCP Systemconfigurator Ethernet Driver

12. Enter the control's IP Address and click the ***Save*** button. You can verify your selections if you have an active connection to the control by clicking the ***Ping*** button.

Note: VisualMotion controls do not support DNS (***D****omain* ***N****ame* ***S****ystem*). Entering a computer name instead of an IP address will produce a successful ping test. However, attempting to communicate with the control will produce an error.

Configuring the PCI Driver

13. Select PCI as the driver type and click the ***Save*** button to add the device.

Modifying a SCP Driver Configuration

A SCP driver configuration can be modified by selecting a device from the tree structure and then clicking the ***Remove Device*** or ***Change Config*** button. Any of the original driver settings can be modified with the exception of the device name. To choose a different device name, the device must first be removed and then added as a new device.

18.4 Remote Access using the SCP Server

SCP remote selection allows the SCP server on one PC to connect to the SCP server on a second PC via DCOM. This connection allows a client PC to access the motion functionality of the control connected to the host PC. Refer to chapter 7, *DCOM Configuration*, in volume 1 of the *VisualMotion Functional Description* manual for details.

Note: SCP remote access is the only method to access the motion functionality of a PPC-P11.1 control hosted on a different PC.

Use the following steps to establish a remote SCP connection to a host PC:

1. From the Windows taskbar, select Start ⇒ Programs ⇒ Rexroth ⇒ SCP ⇒ SCP Remote Selection.
2. Select the computer name of a **Remote Host** or enter the **Remote IP Address** and click the **Apply** button. The window can then be minimized or closed.

Note: Once a connection is established, the SCP server on the host PC is automatically launched, if not currently running.

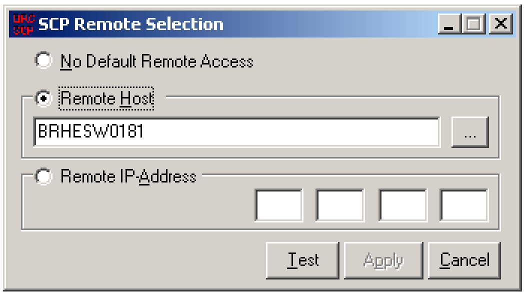


Fig. 18-6: SCP Remote Access

3. With VisualMotion in offline mode, select ***Tools*** ⇒ ***Control Selection***.
4. Select the **SCP (SIS Protocol**) radio button to display to configured SCP targets of the host PC.
5. Switch VisualMotion Toolkit to online mode to establish communication with the control on the remote host PC.

Note: Selecting the **Configure…** button allows the connected PC to make modification to the existing SCP devices.

To disable remote access, reopen the *SCP Remote Selection* window and choose the **No Default Remote Access** radio button.

18.5 OPC Test Client

The OPC Client is a sample client interface that can be used to test the communication link with the SCP server. By referencing the name of the SCP server that was configured using the Systemconfigurator, the OPCClient provides the interface to add groups and items to build the data structure recognized by the SCP server. The data structure hierarchy is composed of groups and items in the format illustrated in Fig. 18-7.

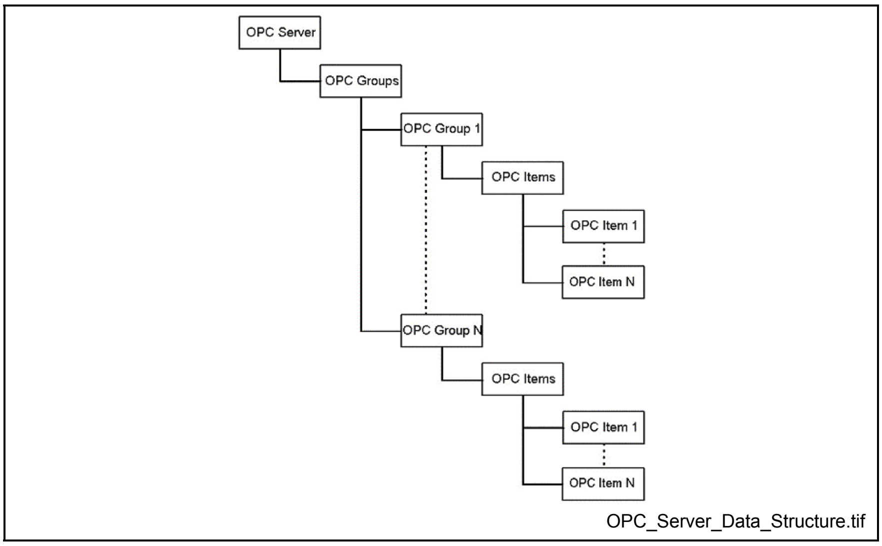


Fig. 18-7: OPC Server Data Structure

To launch the OPC Test Client, select ***Start* ⇒ *Programs* ⇒ *Rexroth* ⇒ *SCP* ⇒ *OPC Test Client***.

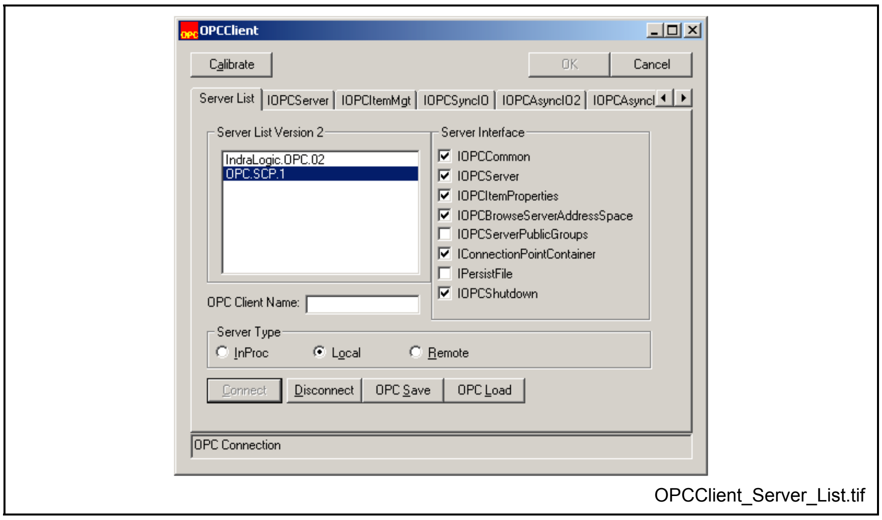


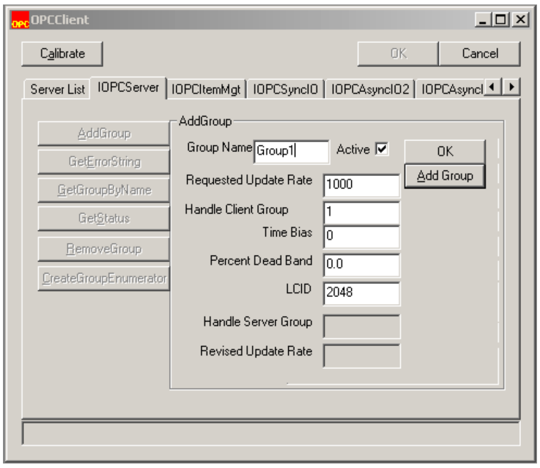
Fig. 18-8: OPC Server List

The server is assigned groups and each group is assigned items. The configuration is not retained when the OPCClient window is closed without storing it on the PC using the ***OPC Save*** button and reloading it using the ***OPC Load*** button in the OPCClient window.

Configure Group

To configure a group:

1. To communicate with a VisualMotion control, select **OPC.SCP.1** from the server list.
2. Click the **Connect** button from the *Server List* tab to establish a connection to the OPC server.
3. Select the *IOPCServer* tab and click **AddGroup**.



OPCClient_IOPCServer.tif

Fig. 18-9: OPC IOPCServer Tab

4. Enter a Group Name, review the default settings of the remaining fields in the tab and change where appropriate.

Note: The value for the Requested Update Rate sets the asynchronous update rate.

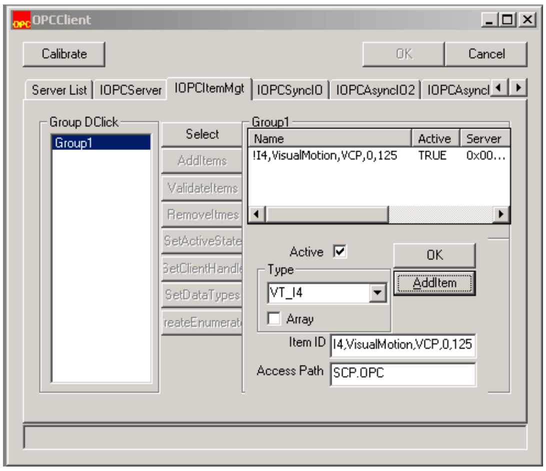
5. Click ***Add Group*** and then ***OK*** to add the group.

Configure Item

To add an item to a group:

1. Select the IOPCItemMgt tab.
2. Highlight the group to add an item to and click the **Select** button.
3. Click the **AddItems** button to activate the item configuration fields.
4. Select the **Type** from the following items in the drop-down menu:
5. Type the **Item ID** and **Access Path** (Access Path = SCP.OPC).

Note: Refer to the OPC Item Structure on page 18-13 for the correct Item ID syntax that is supported by VisualMotion controls.



OPCClient_AddItems.tif

Fig. 18-10: OPCClient IOPCItemMgt Tab

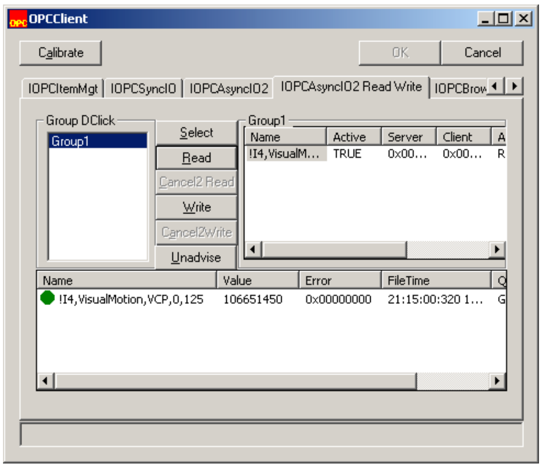
6. Click the **AddItem** button to display the item in the *Group* field.
7. Click the **OK** button when done adding items.

Note: When returning to the IOPCItemMgt tab, existing group items can be viewed by double-clicking on the relevant group name in the *Group Dclick* section or by highlighting the group name and clicking the **Select** button.

The data type can be edited by highlighting the item in the right *Group* field and clicking the **SetDataTypes** button (e.g., VT_I4). To modify any other part of the Item ID, the item must be removed and then re-created.

Read and Write Access

Both Synchronous and Asynchronous read and write access is available to and from the OPC server. Two tabs in the OPC Client window allow you to set the rates and display the results.



OPCClient_IOPCAsyncIO2ReadWrite.tif

Fig. 18-11: IOPCAsyncIO2 Read Write Tab

To activate Asynchronous Read and Write:

1. Highlight the group name and click the **Select** button.

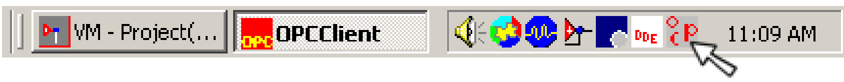
The details of the items in the group are displayed in the group field.

2. Highlight the item and click the **Read** button to obtain the value at the moment the button was clicked.

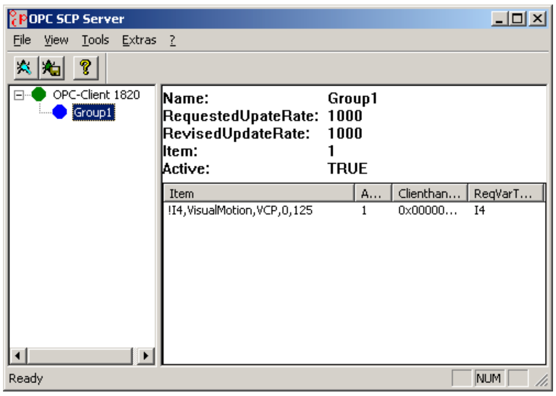
The **Write** button opens a dialog box where a value can be entered for writing to the server. The read and write values are displayed in the field at the bottom of the OPCClient window.

18.6 OPC SCP Server

The hierarchy structure of configured OPC Client groups is displayed in the *OPC SCP Server* window. This window can be launched by clicking on the OPC icon in the Windows startup tray or by opening the OPCScp.exe application file.



Note: The OPC SCP Server only runs while the OPCClient is running.



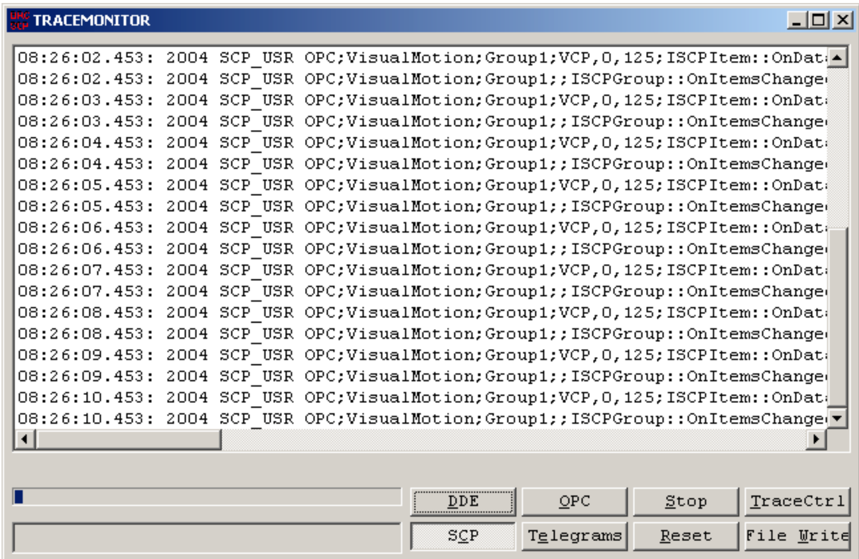
OPC_SCP_Server.tif

Fig. 18-12: OPC SCP Server Window

The number of items configured for the highlighted group is listed in the right window, but the individual items are not displayed.

18.7 TraceMonitor

The TraceMonitor is a Rexroth internal tool for monitoring and debugging SCP communication. The TraceMonitor program is launched by selecting ***Start ⇒ Programs ⇒ Rexroth ⇒ SCP ⇒ SCP TraceMonitor***. It is possible to monitor and analyze the data exchange between the SCP and client applications (such as VisualMotion and Visual Basic clients).



OPC_Tracemonitor.tif

Fig. 18-13: OPC TraceMonitor

18.8 OPC Server Specifications

Specifications The OPC server portion of the SCP supports the following interfaces:

- Data Access Custom Interface Standard Version 2.04 (September 5, 2000)
- Data Access Automation Interface Standard Version 2.02 (February 4, 1999) and OPC Common Definitions and Interface Version 1.0 (October 27,1998) with Release 01V01.

Note: The following interfaces are currently not supported:

- Browsing of available items
- Tree namespace
- Public groups
- OPC Security
- Alarms and Events
- Historical Data Access

Server Types

The following server types are available:

- Local InProc Server
- Local OutProc Server
- Remote Server

Interfaces

The following tables contain the interfaces supported by the OPC server. Additional information about the interfaces is available in the OPC specification document. The command field column in the table indicates the related chapter in the OPC specification.

Interface	Method	Optional	Supported	Comment
IOPCServer	AddGroup		Yes	
	GetErrorString		Yes	
	GetGroupByName		Yes	
	GetStatus		Yes	
	RemoveGroup		Yes	
	CreateGroupEnumerator		Yes	
IOPCServer PublicGroups		Yes	No	
IOPCBrowseServer AddressSpace		Yes	partial	The interface supports just „FLAT Space“
	QueryOrganization		Yes	
	ChangeBrowsePosition	Yes	No	Is not supported because it is just “FLAT Space” supported
	BrowseOPCItemIDs		No	
	GetItemID BrowseAccessPaths	Yes	No	This interface is not supported because there are no AccessPath supported, see also IOPCItemMgt
IOPCItemProperties				New in Version 2.0 of the OPC Specification
	QueryAvailableProperties		Yes	
	GetItemProperties		Yes	
	LookupItemIDs		Yes	
IOPCCommon	SetLocaleID	Yes	Yes	New in Version 2.0 of the OPC Specification
	GetLocaleID		Yes	
	QueryAvailableLocaleIDs		Yes	
	GetErrorString		Yes	
	SetClientName		Yes	
IConnectionPoin Container				New in Version 2.0 of the OPC Specification ConnectPointContainer of the IOPCShutdown interface
	EnumConnectionPoints		Yes	
	FindConnectionPoint		Yes	
IPersistFile		Yes	No	

Table 18-5: OPCServer–Object Interfaces

Interface	Method	Optional	Supported	Comment
IOPCGroupStateMgt	GetState		Yes	
	SetState		Yes	
	SetName		Yes	
	CloneGroup		Yes	
IOPCPublicGroup StateMgt		Yes	No	
IOPCASyncIO2 New in Version 2.0 (OPC Specification)	Read		Yes	
	Write		Yes	
	Refresh2		Yes	
	Cancel2		Yes	
	SetEnable		Yes	
	GetEnable		Yes	
IOPCAsyncIO		Yes	No	Just in Version 1.0 of the OPC Specification
IOPCItemMgt	AddItems		Yes	
	ValidateItems		Yes	
	RemoveItems		Yes	
	SetActiveState		Yes	
	SetClientHandles		Yes	
	SetDatatypes		Yes	
	CreateEnumerator		Yes	
IConnectionPoint Container				New in Version 2.0 of the OPC Specification ConnectPointContainer of the IOPCDataCallback
	EnumConnectionPoints		Yes	
	FindConnectionPoint		Yes	
IOPCSyncIO	Read		Yes	
	Write		Yes	
IDataObject		Yes	No	
IEnumOPCItem Attributes	Next		Yes	
	Skip		Yes	
	Reset		Yes	
	Clone		Yes	

Table 18-6: Interfaces and Methods of the OPCGroup

18.9 OPC Item Structure

The OPC item structure is made up of the following three components:

Item	Description
COM Data Type	The canonical data type that specifies the type and size of data to be accessed. Refer to Table 18-8: COM Data Types for supported types.
Device Name	The local connection name created in the SCP Systemconfigurator for the relevant device. Refer to Device Name on page 18-15 for details.
SCP Command	The SCP command syntax used to access specific motion control data. Refer to SCP Command Syntax on page 18-15 for details

Table 18-7: OPC Item Structure

The complete OPC Item Structure used as an OPC Item ID structure is as follows:

COM Data Type, Device Name, SCP Command

For example:

```
!I4,VisualMotion,VCP,0,125
 |       |           |
 |       |           |__ SCP Command
 |       |______________ Device Name
 |______________________ COM Data Type
```

Note: No spaces can exists in the entire structure and the syntax is case sensitive.

COM Data Type

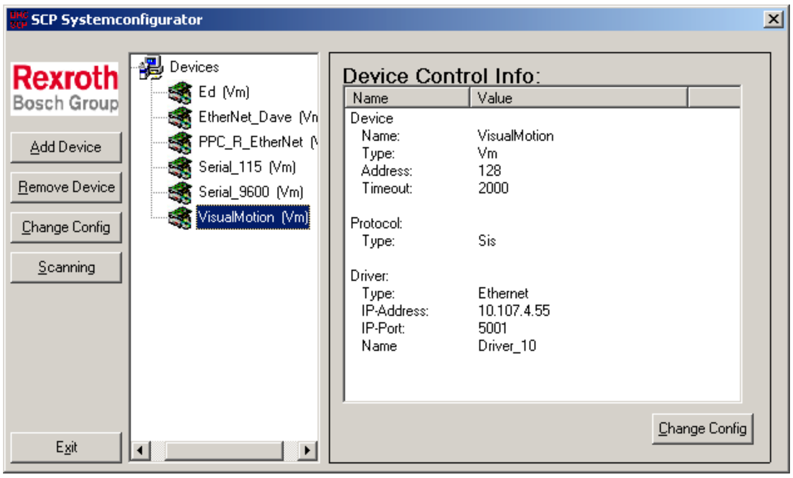
The COM data type is the request data type specifying the type and size of data to be accessed. When requesting an OPC item, the COM data type must start with an exclamation point followed by the data type to be requested. The following table list the supported COM data types:

COM Data Type	Description	Item ID Prefix
VT_EMPTY	The data will be returned as an object. (Canonical Data Type)	!EMPTY
VT_I2	The data will be returned as a signed 16 bit integer.	!I2
VT_I4	The data will be returned as a signed 32 bit integer.	!I4
VT_R4	The data will be returned as a single precision 32 bit floating point number complying with the IEEE 754 standard.	!R4
VT_UI2	The data will be returned as an unsigned 16 bit integer.	!UI2
VT_UI4	The data will be returned as an unsigned 32 bit integer.	!UI4
VT_BOOL	The data will be returned as a Boolean value.	!BOOL
VT_BSTR	The data will be returned as string.	!BSTR
VT_DATE	The data will be returned as a date time value.	!DATE

Table 18-8: COM Data Types

Device Name

The SCP device name identifies the local connection existing in the SCP Systemconfigurator. It can be launched from the Windows taskbar by selecting ***Start ⇒ Programs ⇒ Rexroth ⇒ SCP ⇒ SCP Configurator***. Once opened, all existing configured devices are listed. The name used as the device name in the OPC item structure must match the relevant device to be accessed.



SCP_Systemconfigurator.tif

Fig. 18-14: SPC Systemconfigurator

SCP Command Syntax

The SCP command syntax identifies the actual data that will be accessed from the device. The SCP Command Syntax for the most common commands (parameter, variable, register, event, points, program, control, PID and zone) are defined in the following sections.

Parameter

SCP Command Syntax **VX_1X_2,X_3,X_4,X_5**

V VisualMotion

X_1 Parameter type (D, C, T, or A)

X_2 Data type (A, B, D, G, H, L, P, S, T, U)

X_3 Parameter set number - the drive, axis, or task number depending on the parameter type accessed. The parameter set is always zero for card parameters.

X_4 Parameter number (integer format) – the S-parameter numbers start at zero. The **P** parameters numbers have an **offset of 32768** (for example, the parameter number for P-0-0001 is 32769).

X_5 Number of elements (integer format).

Note: List parameter access information is not available at this time. This information will be available at a later date.

Parameter Type	Data Type
D - Drive Parameter (S and P)	A - Attributes (hex)
C - Card Parameter	B - Block transfer for list parameters
T - Task Parameter	D - Single transfer for list parameters
A - Axis Parameter	H - Upper limit (mixed)
	L - Lower limit (mixed)
	P - Data (mixed)
	S - Parameter Command Status
	T - Parameter name (string)
	U - Unit text (string)

Table 18-9: Parameter SCP Command Syntax

Examples

VisualMotion,VDL,1,125 Access the lower limit of drive parameter S-0-0125, of Drive number 1, from the device “VisualMotion”

Event

SCP Command Syntax

VEX$_1$,X$_2$,X$_3$

V VisualMotion

E Event table

X_1 Data type (S, T, D,A, F,M, or L)

X_2 Program number in decimal format (0 indicates active program)

X_3 Event number 1 to n in decimal format (n is determined through the offline VisualMotion data table)

Data Type	Description
S	Status, integer data
T	Type, integer data
D	Direction, integer data
A	Argument, float data
F	Function, string data
M	Message, string data
L	Array format, string data

Table 18-10: Event SCP Command Syntax

Examples

VisualMotion,VEL,0,1 Access event entry (program # = 0, Event = 1)

VisualMotion,VEM,0,1 Access message of event 1 of the active program

Jog

SCP Command Syntax

VJX_1,X_2,X_3,X_4

V VisualMotion

J Jog Function

X_1 Data type (A, B, or C)

X_2 Program number in decimal format (0 indicates the active program) – used only for labels

X_3 Register number in decimal format

X_4 Bit number in decimal format

Data Type	Data Value	Description
A	1	Jog Axis with No Error Detection
B	1	Jog Axis with Error Detection
C	0	Cancel Jog Immediately

Table 18-11: Jog SCP Command Syntax

Examples

VisualMotion,VJA,0,11,2 Jog Axis 1 with No Error Detection (Program # = 0, Register number = 11, Bit = 2, State = 1)

VisualMotion,VJB,0,11,2 Jog Axis 1 with Error Detection (Program # = 0, Register number = 11, Bit = 2, State = 1)

VisualMotion,VJC,0,11,2 Cancel Jog Immediately (Program # = 0, Register number = 11, Bit = 2, State = 0)

Refer to section 7.5, *Jogging*, in volume 1 for detailed information.

Variable

SCP Command Syntax

V$X_1$$X_2$,$X_3$,$X_4$

V VisualMotion

X_1 Variable type (F, I, H, or G)

X_2 Data type (P, X, or T)

X_3 Program number in decimal format (0 indicates active program)

X_4 Variable number 1 to n in decimal format (n is determined through the offline VisualMotion data table)

Variable Type	Description
F	Program Floats
I	Program Integers
H	Global Floats
G	Global Integers
Data Type	
P	Float/Integer data
X	Hex data
T	Label text

Table 18-12: Variable SCP Command Syntax

Examples

VisualMotion,VFP,0,1 Access floating-point F1 variable of the active program.

Register

SCP Command Syntax

VRX$_1$,X$_2$,X$_3$

V VisualMotion

R Register

X_1 Data type

X_2 Program number in decimal format (0 indicates the active program) – used only for labels

X_3 Register number in decimal format

Data Type	Description
B	Binary data
D	Decimal data
X	Hex data
M	Hex set bits
F	Binary force mask
S	Binary force state
C	Hex force mask and state
E	Clear all forcing
T	Label text

Table 18-13: Register SCP Command Syntax

Examples

VisualMotion,VRD,0,100 Access register state (Program # = 0, Register number = 100)

VisualMotion,VRB,0,100 Access Register 100 of the currently active program and display the data in binary format.

Point

SCP Command Syntax

VX$_1$X$_2$,X$_3$,X$_4$

V VisualMotion

X_1 Point type

X_2 Data type

X_3 Program number in decimal format (0 indicates active program)

X_4 Point number 1 to n in decimal format (n is determined through the offline VisualMotion data table)

Point Type	Description
X	Absolute Points ABS
Y	Relative Points REL

Data Type	Description
X	X coordinate (float data)
Y	Y coordinate (float data)
Z	Z coordinate (float data)
B	Blend radius (float data)
S	% of Maximum speed (integer data)
A	% of maximum acceleration (integer data)
D	% of maximum deceleration (integer data)

Data Type	Description
J	% jerk (integer data)
1	Event number to enable for this point (integer data)
2	Event number to enable for this point (integer data)
3	Event number to enable for this point (integer data)
4	Event number to enable for this point (integer data)
R	Roll/Rate (float data)
P	Pitch (float data)
W	Yaw (float data)
E	Elbow state (integer data)
L	Array format (string) data
V	Label list (string)

Table 18-14: Point SCP Command Syntax

Examples

VisualMotion,VXL,0,1 Access absolute point 1 from the active program.

VisualMotion,VXY,0,10 Access the y-coordinate of absolute point 10 from the currently active program.

Program

SCP Command Syntax

VPX$_1$,X$_2$,X$_3$

V VisualMotion

P Program

X_1 Subclass

X_2 Program Number

X_3 Set

Subclass	Description	Program Number	Set
A	Activate a program (integer data)	0	0
C	Clear all programs	0	0
D	Trnasfer program data (string)	The actual program number	Element of prog download
E	Delete a program (integer data)	0	0
G	Coordinated Articulation block variable	0	Coord. Art. block numer
H	List of programs (string)	0	The element of the list
J	CAM indexer (string)	0	CAM ID block number
K	Compress flash (integer data)	0	0
M	Registration block information (string)	0	Registeration block number
N	Program name (string)	The actual program number	0
R	Upload porgram header (string)	0	0
T	Transfer data between programs	Source program number (0)	Destination program number
V	List of program variables (string)	The actual program number	0
W	Download program header (string)	0	0
X	Copy program data to another program (integer data)	The number of the source program	0

Table 18-15: Program SCP Command Syntax

Examples

VisualMotion,VPN,10,0 Accesses the name of program 10.

Writing the program number to **VisualMotion,VPE,0,0** deletes the program.

Writing the program number to **VisualMotion,VPA,0,0** activates the program.

Writing the program number n to **VisualMotion,VPX,4,0** copies the program data from program n to program 4.

Control PLS

SCP Command Syntax

VWX$_1$,X$_2$,X$_3$

V VisualMotion

W Control PLS

X_1 Data type

X2 Program number – the number of the program from which data is being accessed (0 is the currently active program).

X3 PLS number

Data Type	Description
O	Phase offset (float data)
R	Assigned output register (integer data)
M	Assigned mask register (integer data)
T	Master type (integer data)
A	Master axis number (integer data)
H	List of On, Off, and Lead time values (string)
E	List of On values (float data)
F	List of Off values (float data)
G	List of lead-time values (integer data)

Table 18-16: Control PLS SCP Command Syntax

Examples

VisualMotion,VWM,1,2 Access the assigned mask register of control PLS of program 1.

PID

SCP Command Syntax

VMX$_1$,X$_2$, X$_3$

V VisualMotion

M PID

X_1 Data type

X_2 Program number – the number of the program from which data is being accessed (0 is the currently active program).

X_3 PID number

Data Type	Description
B	Variables used (string data)
E	Calculated set point (float data)
F	Calculated feedback (float data)
G	Calculated output (float data)
J	Loop time (integer data)
L	List of valid PID loop numbers (integer data)
R	Control register (integer data)
S	Status register (integer data)
T	Type (integer data)

Table 18-17: PID SCP Command Syntax

Examples

VisualMotion,VMS,0,1 Access the number of the status register of PID loop 1 of the currently active program.

Zone

SCP Command Syntax

VZX_1,X_2,X_3

V VisualMotion

Z Zone

X_1 Data type

X_2 Program number 0 to 10 in decimal format (0 indicates current program)

X_3 Zone number 1 to n in decimal format (n is determined through the offline VisualMotion data table)

Data Type	Description
S	Status (integer data)
A	Point 1 X coordinate (float data)
B	Point 1 Y coordinate (float data)
C	Point 1 Z coordinate (float data)
D	Point 2 X coordinate (float data)
E	Point 2 Y coordinate (float data)
F	Point 2 Z coordinate (float data)
L	Array format (string data)

Table 18-18: Zone SCP Command Syntax

Examples

VisualMotion,VZL,0,1 Access zone 1 array of the active program.

VisualMotion,VZA,0,3 Access the x-coordinate from zone 3 of the currently active program.

18.10 Sample OPC Clients

Examples of the Wonderware and VisualBasic clients can be installed with VisualMotion if a custom installation is performed and the ***OPC Sample Applications*** check box is selected. Reference documents for these samples:

- OPC Client Information in WinStudio
- OPC Client Information in Wonderware 7
- OPC Client Sample for .Net
- OPC Client Sample in VisualBasic6

are installed in the file location:

C:\…\Rexroth\VisualMotion11\SCP\Samples

These documents are provided in PDF format and contain information about source code, establishing communication to the OPC server, and reading and writing data to and from the OPC server.

18.11 Dde Communication for SCP

The SCP supports the Dde communication protocol so that a VisualMotion 11 project can communicate over a network, serial, or PCI connection. SCP also provides HMI support of Dde communication. The protocol for the SCP-Dde server uses the same ASCII syntax that the VM Dde server uses. To establish communication, a Service, Topic, and Item name need to be specified. Their formats are:

Service Name:	*CLC_Dde*. **(for exchanging control data)**
Topic Name:	Device name defined in Systemconfigurator
Item Name:	**Direct ASCII serial protocol** (identify the data to be exchanged.)

Refer to the *VisualMotion 9 Application* manual or the *VisualMotion 11 Help System* for Direct ASCII communication details.

19 Communication Protocols

19.1 Protocol Overview

VisualMotion controls can receive and send data by means of a serial interface, network or PCI-Bus (PPC-P11.1) connection. A variety of devices and programs can communicate with the PPC-P11.1 and PPC-R22.1. Both controls support binary SIS (standard Rexroth protocol) protocol, described in this chapter, or ASCII protocol.

VisualMotion controls can be ordered with a serial and/or Ethernet communication port. Communication between VisualMotion Toolkit and the control is accomplished across these ports using the Scalable Communication Platform (SCP) or the VisualMotion Dde (VM Dde) server. By means of these communication platforms, the user can perform the following functions:

- edit system and drive parameters,
- edit data types, such as variables, point tables, etc.
- download user programs,
- provide diagnostic information,
- etc

Note: The VisualMotion Dde server only supports communication between VisualMotion Toolkit and the control (motion functionality). IndraLogic communication requires the SCP server (SIS protocol).

Serial Interface Mode

The RS-232, RS-422 and RS-485 serial communication interface modes are supported.

Features

- Transmission rate: 9600 through 115200 baud rate
  - Maximum transmission path: 15 meters (approx. 50 ft) for RS232 and RS485
- 8 bit ASCII protocol
- no parity
- a stop bit

Network Mode

The Ethernet interface is used to transmit data over a local area network or Internet.

Features

- Transmission modes: half and full duplex
  - Transfer rate: up to 100 Mbps (**M**ega**b**its **p**er **s**econd)
- Communication Protocol: TCP/IP
- Cable type: RJ-45

PCI Mode

The PCI interface is only available for a PPC-P11.1 control installed in an Industrial PC or desktop PC.

Rexroth
Bosch Group

19.2 Communication Protocols

VisualMotion 11 supports the following protocol:

- SIS Protocol
  (standard Rexroth binary protocol using a Scalable Communication Platform SCP)
- ASCII Protocol
  (VisualMotion Dynamic Data Exchange server)

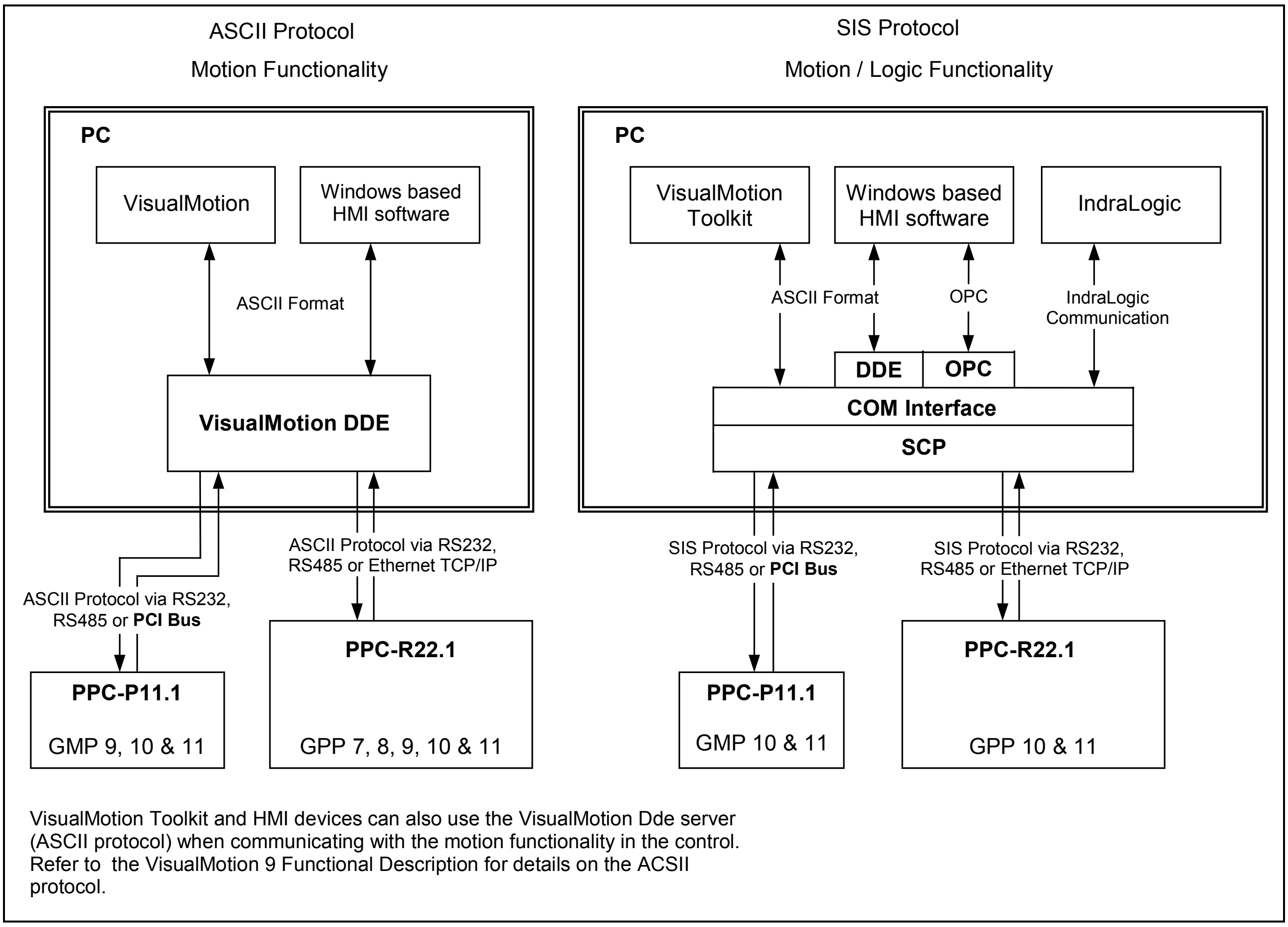


Fig. 19-1: SIS Communication Overview

SIS Protocol

The SIS protocol defines a peer-to-peer network where any node can be a client or server. Applications can be implemented to perform routing duties for hierarchical networks. SIS uses binary messages transported in telegram frames between a client and server for reading and writing data. Data such as registers, floating point and integer variables, control, axis, or task parameters, as well as drive parameters over various physical media, including EIA-232, EIA-485, and 10Base-T.

Features:

- Binary protocol
- A checksum test is conducted (higher Hamming distance D)
- All telegrams are identified by an unequivocal start symbol
- Defined telegram frame structure

Structure and Telegram frame

The telegrams used for data exchange over SIS is structured as follows:

- Telegram header
- User Data header (dependents on SIS service)
- User data

Communication Parameters

The data exchange is controlled by means of the following parameters:

- **C-0-0003**, baud rate and parity setup for X10
- **C-0-0004**, baud rate and parity setup for X16
- **C-0-0005**, enable communication protocol
- **C-0-0002**, device address

19.3 SIS Communication

Note: This section only covers basic information on Common and GPP-Specific SIS services. Refer to the information manual, "DOK-GENERL-SIS*DEFINIT-IF02-EN-P" material number 289718, for detailed information and examples.

Telegrams

The telegrams used for data exchange over SIS are generally structured according to the following scheme:

- Telegram header
- User data header (depends on the SIS service)
- User data

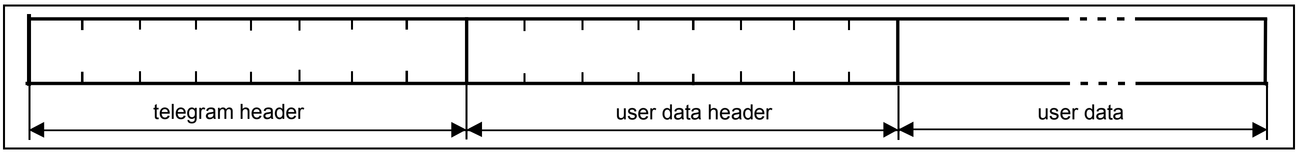


Fig. 19-2: Telegram Structure for Data Exchange via SIS

However, the exact structure of the individual telegrams depends not only on the SIS service but also on the telegram direction and the telegram type.

Telegram Directions and Types

Telegram Directions

The following two telegram directions are supported over SIS.

Telegram Direction	Sender of the Telegram
Command telegram	Master: the 'active' communications partner
Response telegram	Slave: the 'passive' communications partner

Table 19-1: Telegram Directions

Theoretically, with a RS485 connection every bus user can be both a master - i.e. can send a command telegram that is then answered by the slave with a response telegram – as well as a slave, simultaneously.

However, in practice there is a distinction between master partners (control devices) and slave partners (target devices).

Telegram Types in the Command Telegram

There are four types of command telegrams for the master:

Telegram Type	Data Direction
SEND telegram	Write access: data sent to a slave
RETRIEVE telegram	Read access: data requested from a slave
Group message	Message: data sent to a group of slaves
Broadcast message	Message: data sent to all slaves

Table 19-2: Telegram Types in the Command Telegram

Note: Group and broadcast messages are **not** answered by the slave's address.

Telegram Types in the Response Telegram

As appropriate, a slave that is addressed at its own address sends a response telegram as follows.

Telegram Content	Telegram Type
Transmission status	with error-free SEND telegram
Transmission status and requested data	with error-free RETRIEVE telegram
Transmission status and error code	with erroneous SEND or RETRIEVE telegram

Table 19-3: Telegram Content of the Response Telegram

Sequential Telegrams

Long data sets in the various forms of telegram directions and types may have to be subdivided among several subtelegrams and sent as sequential telegrams.

Structure of the Command Telegram

Write Access

A SEND telegram is made up:

- Telegram header
- User data header (depends on the SIS service)
- User data

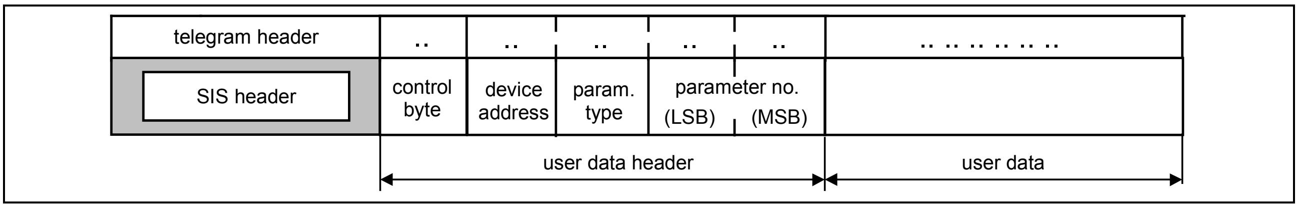


Fig. 19-3: Structure of the Command Telegram: Write Access

Read Access

A RETRIEVE telegram is made up of:

- Telegram header
- User data header (depends on the SIS service)

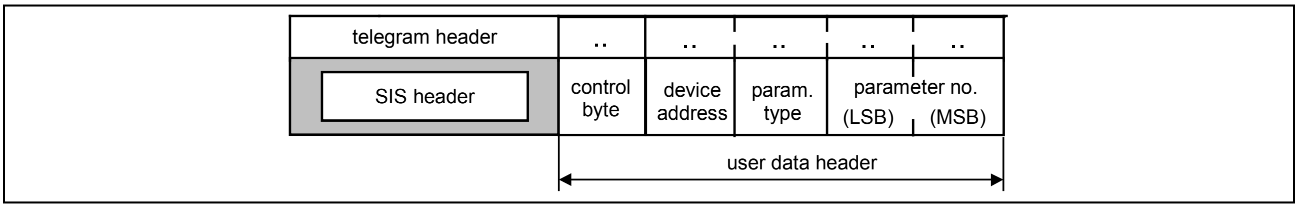


Fig. 19-4: Structure of the Command Telegram: Read Access

The user data header for specifying the read access can also be omitted if it is already set in the telegram header by the SIS service.

Structure of the Response Telegram

Write Access

The response telegram is made up:

- Telegram header
- User data header (depends on the SIS service)
- Error code, if applicable

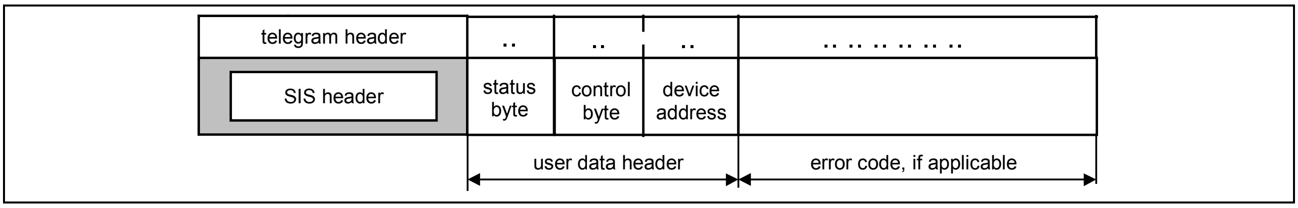


Fig. 19-5: Structure of the Response Telegram: Write Access

Read Access

The response telegram is made up of:

- Telegram header
- User data header (depends on the SIS service)
- User data or error code

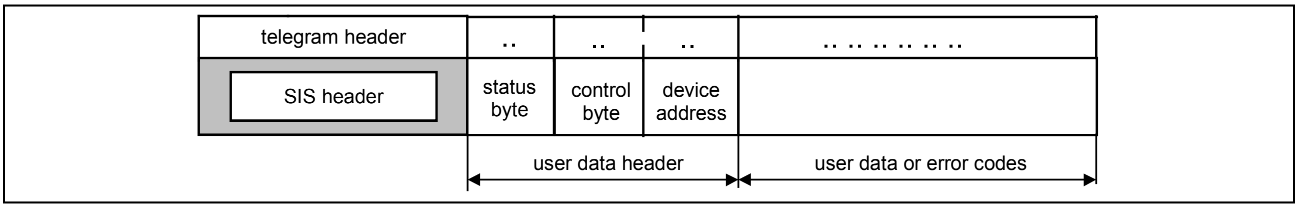


Fig. 19-6: Structure of the Response Telegram: Read Access

The response telegram from the slave has, apart from a few differences, the same telegram and the same user data header as the command telegram. This enables the sender of a command telegram to assign a unique response telegram.

Differences in the telegram header:

In the telegram header, the telegram must be marked as a response telegram. Refer to DOK-GENERL-SIS*DEFINIT-IF02-EN-P for details.

Differences in the user data header:

A status byte for displaying the transmission status is shown in the first byte of the user data header for general SIS services.

Note: The user data depends on the SIS service and the status byte.

Data Formats in the Telegrams

The SIS telegram is a **binary telegram** with standardized, binary telegram header and content for all general SIS services.

Note: The telegram content may also consist of an ASCII data set for the special SIS services in the individual product groups.

Intel Format

The allocation of the individual bytes of data of the 'Word' or 'DWord' type corresponds to the **Intel** convention.

Example:

In a defined data structure, the data words 0x0008 and 0x001F as well as the double word 0x1D000164 should be transmitted.

The logical order of the data:

00	08	00	1F	00	02	1D	00	01	64
Word		Word		Byte	Byte	Long (Double Word)			

The order of bytes sent in **Intel-Format**:

08	00	1F	00	00	02	64	01	00	1D

IEEE Format

The **IEEE** format applies to the float display of data.

Telegram Header

The telegram header contains the following elements used for telegram control.

- Payload length
- Subnet routing
- Data integrity
- Service identification
- Source and destination addressing

Bits 0 - 3 in the telegram header control byte

In addition to general technical transmission control, it must also meet the following specific requirements:

- Addressing with up to four sub-addresses
- Forwarding of the telegram to the next recipient in the simplest possible way (the existence of devices that cannot process such telegrams but will have to forward them should be taken into consideration).

- Data in the telegram header that may have to be changed or evaluated when forwarding a telegram should be in a fixed position in the telegram header.

SIS Header Service Identification

Byte 6 of the SIS header is used for specifying a SIS service. GPP supports the following specific services:

- General services
- Common services
- GPP-Specific services

General SIS Services

Service ID	Function
0x00	User Identification
0x03	Initialize SIS communication

Table 19-4: General SIS Services

Common SIS Services

Service ID	Function
0x10	Read parameter
0x11	Read list parameter
0x1E	Write list parameter
0x1F	Write parameter

Table 19-5: Common SIS Services

GPP-Specific SIS Services

Service ID	Function
0xC3	Read data access
0xC4	Write data access

Table 19-6: GPP-Specific SIS Services

User Data Header

The content structure of the user data header dependents on the specific common and GPP-Specific SIS services that are initiated in the SIS header. GPP supports the following:

- Common parameter command / response telegrams
- Common list segment command / response telegrams
- GPP-Specific command / response parameter telegrams

Parameter Command / Response Telegrams

For read or write parameters that are listed as common, the command and status telegram formats are described in the figure below. Read operations use the “0x10” SIS service, whereas write operations use the “0x1F” SIS service.

Read / Write Parameter Command Telegram

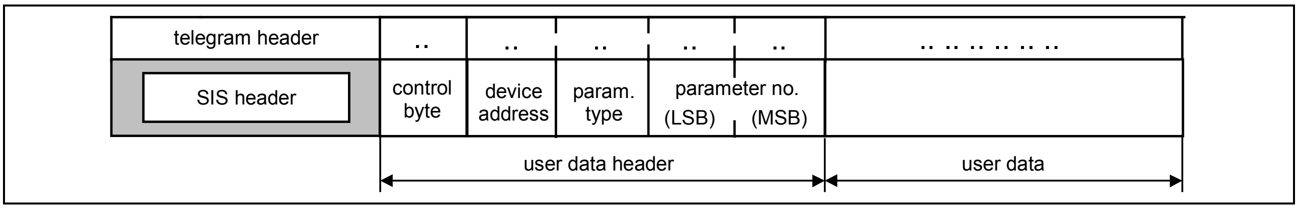


Fig. 19-7: Read / Write Parameter Command Telegram

Read / Write Parameter Response Telegram

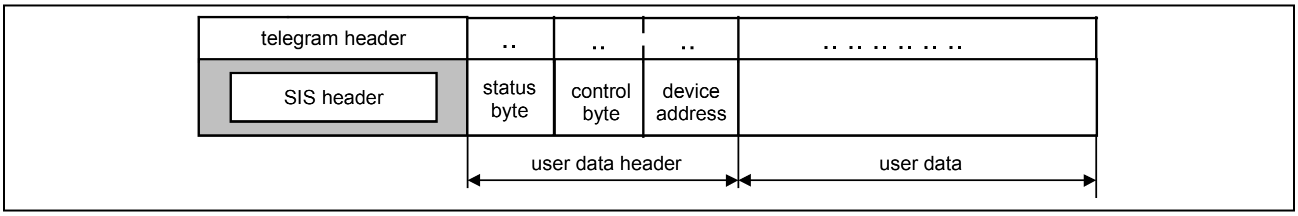


Fig. 19-8: Read / Write Parameter Response Telegram

Parameter List Segment Command / Response Telegrams

For read or write list parameters that are listed as common, the command and status telegram formats are described in the figure below. Read operations use the “0x11” SIS service, whereas write operations use the “0x1E” SIS service.

Read / Write List Segment Command Telegram

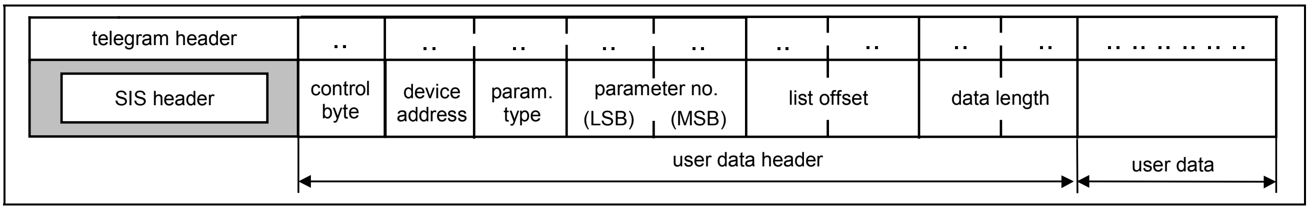


Fig. 19-9: Read / Write Parameter List Segment Command Telegram

Read / Write List Segment Response Telegram

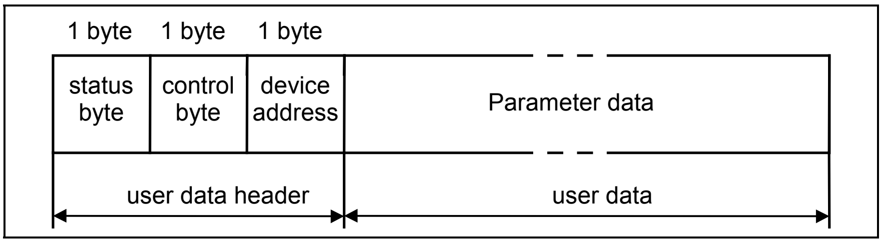


Fig. 19-10: Read / Write Parameter List Segment Response Telegram

Control Byte for Common SIS Service

The following user data header control byte is used for common SIS services. The control byte specifies how a data block element of a parameter is accessed. The transmission of a consecutive telegram is controlled with bit 2.

The control byte is read out of the command telegram and copied into the response telegram. The transmission of a consecutive telegram is controlled with bit 2 (lists read in several steps). Refer to SIS Header Service Identification on page 19-7 for details.

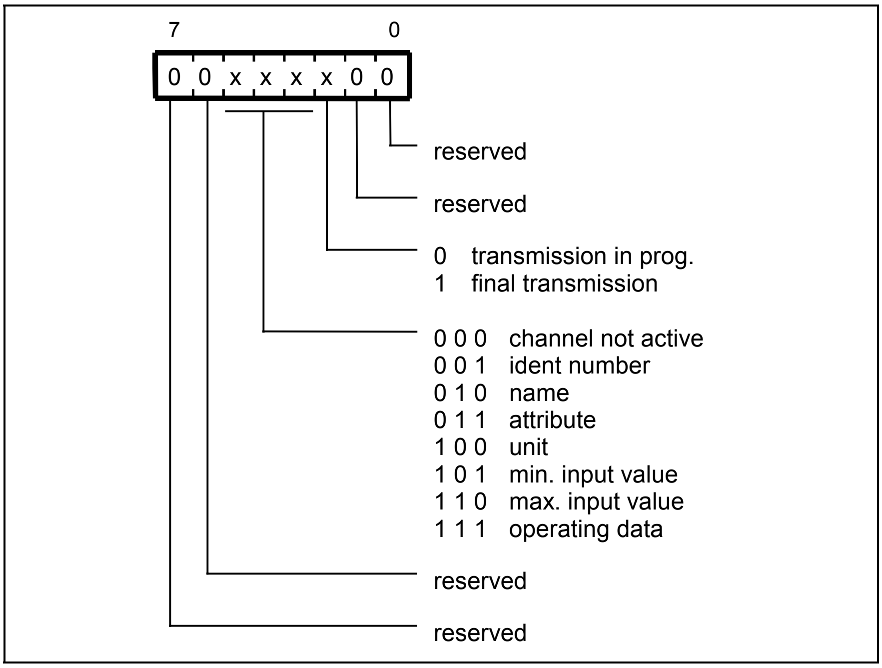


Fig. 19-11: Read / Write Parameter Command / Response Control Byte

Device Address

The unit address of a drive is read in the command telegram and copied into the response telegram.

The SIS protocol permits:

- direct SIS communication with drives supporting SIS interface. In this case the unit address is the same as the SIS address of the receiver.
- accessing drive parameters via a motion control, in case of drives not supporting SIS interface. The SIS address is related to the motion control and the unit address to the drive.

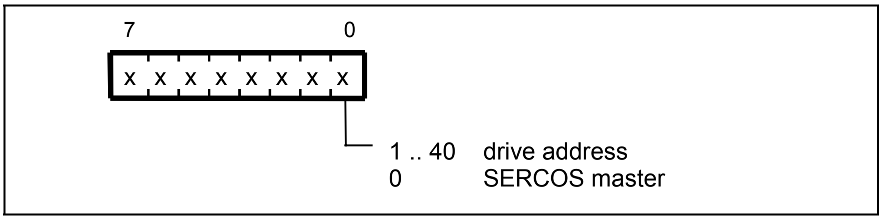


Fig. 19-12: Device Address

Parameter Type and Number

The parameter number has the form as defined in the SERCOS interface specification. To be able to also address control parameters, one byte is set ahead of the address to identify the parameter type.

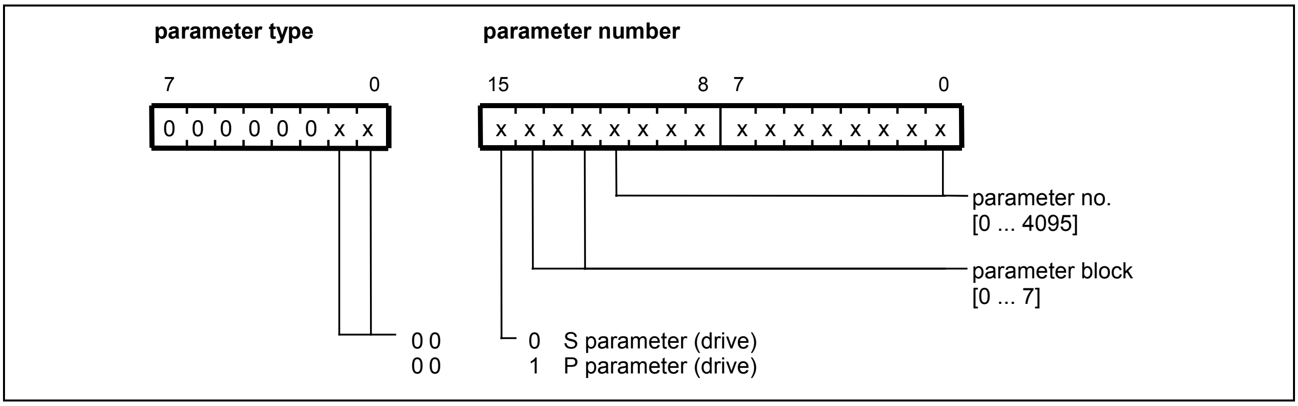


Fig. 19-13: Parameter Identification

Status Byte Error Codes

The status byte of the user data header provides the result of the transmission in the form of an error code. Generally, status codes can be separated into the following three types.

Error type	Error code
Transmission error-free	0x00
Protocol error	0xF0 ... 0xFF
Execution error (see below)	0x01 ... 0xEF

Table 19-7: Error types

The protocol error codes are defined as follows:

Code No.	Description	Error Type
0xF0	"Invalid service": The requested service is not specified or is not supported by the addressed user.	Telegram error
0xF1	"Invalid telegram": The telegram cannot be evaluated because, for example, a slave received a response telegram from the master or the start character was not found.	Telegram error
0xF2	"Telegram length error": The two length entries in the telegram do not match.	Telegram error
0xF4	"Checksum error": The transmitted checksum does not match the one calculated internally.	Telegram error
0xF8	"Invalid sequential telegram": Data in the user data header, the sender address or the service has changed in the sequential telegram.	Telegram error

Table 19-8: Response Telegram Status Byte Protocol Errors

Execution error codes are defined as follows:

Execution error	Code No.	Description
"Error during parameter transmission"	0x01	An error occurred while reading or writing a parameter (see below "Error codes in SIS services 0x10 – 0x1F")
"Error during phase switching"	0x02	The specified target phase was not achieved (see below " Error codes in SIS services 0x10 – 0x1F")

Table 19-9: Execution Error Codes

GPP-specific Command / Response Parameter Telegrams

The VisualMotion ACSII protocol is mapped to the SIS protocol and uses the command and response formats described in this section.

For read or write parameters that are listed as GPP-Specific, the command and status telegrams are described in the figure below. Read operations use the “0xC3” SIS service, whereas write operations use the “0xC4” SIS service.

Read / Write Parameter Command Telegram

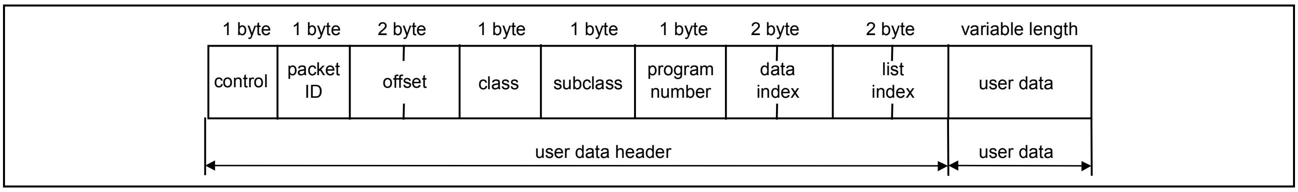


Fig. 19-14: GPP-specific Read / Write Parameter Command Telegram

User Data Header Element	**Description**
Control	Contains message fragment information and is used in conjunction with the “**Packet ID**” byte (see below). Bit 2 indicates whether or not the message is fragmented. When fragmented, bit 2 is set to 0 and the “**Packet ID**” indicates the fragment number in ascending order (e.g. 1—N). This allows GPP to reassemble the message if frames are received out of order, and when there are differing MTU sizes. The last fragment has bit 2 set to 1. When there are no fragments, bit 2 is set to 1 and the “**Packet ID**” is 1. Refer to section “GPP-Specific Read / Write Parameter Command / Response Control Byte” for bit definitions.
Packet ID	Number indicating the message fragment number (see control byte description)
Offset	Starting location for the return data
Class	Data type such as a register, user program, cam, or variable. Refer to Table 19-12 to Table 19-15.
Subclass	Requested format, element, or operation. Refer to Table 19-12 to Table 19-15.
Program Number	Indicates user-program, drive, task, or axis number (binary field). 1. A user program is identified by 0--10, where “0” indicates the active program. 2. A drive parameter number is selected by its SERCOS number (1--40). 3. An axis parameter number corresponds to its drive number (1--40). 4. A task parameter is identified as 1 for Task A, 2 for Task B, 3 for Task C, and 4 for Task D.
Data Index	Indicates a variable index, parameter, block, or step number (binary field). 1. Indicates variable number from 1 to the total number of variables available. A “0” requests the number of variables. 2. Indicates the parameter number. 3. Indicates the block number during a program transfer. 4. Indicates the step number for the sequencer.
List Index	Requests a specific item number in a list parameter, or the step number. A “0xFFFF” indicates this field is unused (binary field)
User Data	Contains the data read, written, or the operation status (ASCII field)

Table 19-10: GPP-specific Read / Write Parameter Command Telegram Elements

Read / Write Parameter Response Telegram

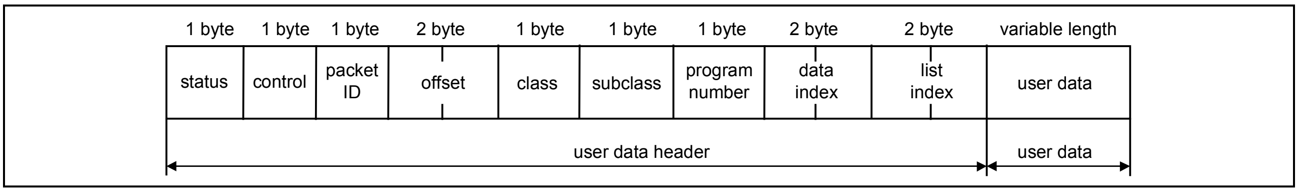


Fig. 19-15: GPP-specific Read / Write Parameter Response Telegram

User Data Header Element	Description
Status	Result of a transmission in the form of a code number.
Control	Same description as command telegram
Packet ID	Same description as command telegram
Offset	Starting location for the return data. Modified by GPP if the return data is larger than the maximum MTU size. The client then resends the command with the modified offset to collect the next fragment.
Class	Same description as command telegram
Subclass	Same description as command telegram
Program Number	Same description as command telegram
Data Index	Same description as command telegram
List Index	Same description as command telegram
User Data	Contains the data read, written, or the operation status (ASCII field).

Table 19-11: GPP-specific Read / Write Parameter Response Telegram Elements

GPP Class and Subclass Types

The following tables contain the SIS protocol class and subclass for accessing VisualMotion GPP data types.

Data Type	Class	Subclass	List	ASCII Format
Registers	0x52	0x42	N	Binary data
		0x44	N	Decimal data
		0x58	N	Hex data
		0x4D (Write Only)	N	Hex set bits
		0x46	N	Binary Force Mask
		0x53	N	Binary Force State
		0x43	N	Hex Force Mask & State
		0x45 (Write Only)	N	Clear all forcing
Program Floats, Program Integers, Global Floats, Global Integers	0x46, 0x49, 0x48, 0x47	0x50	N	Decimal data
		0x58	N	Hex data
		0x4C(Read Only)	N	Label text
Points(ABS, REL)	0x58, 0x59	0x58	N	X coor., float data
		0x59	N	Y coor., float data
		0x5A	N	Z coor., float data
		0x42	N	Blend radius, float data
		0x53	N	% of max. speed, integer data
		0x41	N	% of max. accel., integer data
		0x44	N	% of max. decel., integer data
		0x4A	N	% jerk, integer data
		0x31	N	number of event enabled for this point, integer data
		0x32	N	number of event enabled for this point, integer data
		0x33	N	number of event enabled for this point, integer data
		0x34	N	number of event enabled for this point, integer data
		0x52	N	Roll / Rate, float data
		0x50	N	Pitch, float data
		0x59	N	Yaw, float data
		0x45	N	Elbow state, integer data
		0x4C	N	Array format, string data
		0x54	N	Label text

Table 19-12: GPP Data Types (Registers, Program Variables and Points)

Data Type	Class	Subclass	List	ASCII Format
Zones	0x5A	0x53(Read Only)	N	Status, integer data
		0x41	N	Point 1 X coor., float data
		0x42	N	Point 1 Y coor., float data
		0x43	N	Point 1 Z coor., float data
		0x44	N	Point 2 X coor., float data
		0x45	N	Point 2 Y coor., float data
		0x46	N	Point 2 Z coor., float data
		0x4C	N	Array format, string data
		0x54	N	Label text
Events	0x45	0x53(Read Only)	N	Status, integer data
		0x54	N	Type, integer data
		0x44	N	Direction, integer data
		0x41	N	Argument, float data
		0x46	N	Function, string data
		0x4D 0x4C	N N	Message, string data Array format, string data
		0x4E	N	Label text
PID	0x4D	0x42(Read Only)	N	Variables Used, string data
		0x45(Read Only)	N	Calculated set point, float data
		0x46(Read Only)	N	Calculated feedback, float data
		0x47(Read Only)	N	Calculated output, float data
		0x4A	N	Loop time, integer data
		0x4C(Read Only)	Y	List of valid PID loop numbers, integer data
		0x52(Read Only)	N	Control register, integer data
		0x53(Read Only)	N	Status register, integer data
		0x54(Read Only)	N	Type, integer data
PLS	0x57	0x4F	N	Phase offset, float data
		0x52	N	Assigned output register, integer data
		0x4D	N	Assigned mask register, integer data
		0x54	N	Master type, integer data
		0x41	N	Master axis number, integer data
		0x48	Y	List of On, Off, and Lead time values, string
		0x45	Y	List of On values, float data
		0x46	Y	List of Off values, float data
		0x47	Y	List of Lead-time values, integer data

Table 19-13: GPP Data Types (Zones, Events, PID and PLS)

Data Type	Class	Subclass	List	ASCII Format
Program	0x50	0x41	N	Activate a program, integer data
		0x43	N	Clear all programs
		0x44	Y	Transfer program data, string
		0x45	N	Delete a program, integer data
		0x46(Read Only)	Y	List of program event functions, string
		0x47	N	Coordinated Articulation block variable
		0x48(Read Only)	Y	List of programs, string
		0x4A(Read Only)	Y	List of CAM INDEXER control blocks, string
		0x4B	N	Compress Flash, integer data
		0x4D	Y	List of REGISTRATION control blocks, string
		0x4E(Read Only)	N	Program name, string
		0x52(Read Only)	N	Upload program header, string
		0x56(Read Only)	Y	List of program variables, string
		0x57	N	Download program header, string
		0x54	N	Transfer of data between program
		0x58	N	Copy program data to another program, integer data
ELS	0x4B	0x47(Read Only)	N	ELS GROUP control blocks, string
		0x4C(Read Only)	Y	List of drives assigned to a ELS GROUP
		0x4D(Read Only)	N	ELS MASTER control blocks, string
		0x56(Read Only)	N	VIRTUAL MASTER control blocks, string
Parameters (A, C, D, T)	0x41, 0x43, 0x44, 0x54	0x41(Read Only)	N	Attributes, hex
		0x42	Y	List of data in block mode, mixed
		0x44	Y	List of data one at a time, mixed
		0x48(Read Only)	N	Upper limit, mixed
		0x4C(Read Only)	N	Lower limit, mixed
		0x50	N	Data, mixed
		0x53(Read Only)	N	Parameter command status
		0x54(Read Only)	N	Parameter name, string
		0x55(Read Only)	N	Unit text, string

Table 19-14: GPP Data Types (Programs, ELS and Parameters)

Data Type	Class	Subclass	List	ASCII Format
Functions	0x53	0x41	N	Access type, integer data
		0x48	Y	List of Argument upper limits, mixed
		0x4C	Y	List of Argument lower limits, mixed
		0x4E	Y	List of Argument names, string
		0x52	Y	List of argument arrays, name, type, min, max, string
		0x54	N	Function name, string
		0x56	Y	List of Local Variables arrays, string
		0x59	Y	List of Argument types, char
Sequencer	0x4C	0x54	N	Sequencer name, string
		0x50	Y	List of Sequence table numbers, integer data
Sequencer Table	0x51	0x54	N	Table name, string
		0x50	Y	List of table data, string
		0x46	Y	List of functions in table, integer data
		0x31	Y	List of argument 1, mixed
		0x32	Y	List of argument 2, mixed
		0x33	Y	List of argument 3, mixed
		0x34	Y	List of argument 4, mixed
		0x35	Y	List of argument 5, mixed
Jogging	0x4A	0x41	N	Set register bit and start timer1
		0x42	N	Set register bit and start timer2
		0x43	N	Restore register bit and stop timers

Table 19-15: GPP Data Types (Functions, Sequencer and Jogging)

20 PPC-P11.1 Control Functionality

20.1 Overview

The PPC-P11.1 is a PCI motion/logic control card designed to work with multiple application-specific firmware. This chapter describes how the PPC-P11.1, using GMP 11 firmware, works as part of a complete PLC / Motion Control system solution. The PPC-P11.1 can function as a stand-alone motion control with integrated PLC functionality (IndraLogic) or be combined with additional system components, such as:

- Logic controller (SoftPLC)
- Fieldbus master
- HMI package (WinHMI or WonderWare)
- I/O devices
- Digital drives and motors

Communication between the PPC-P11.1 and system components is performed via a Dual Port RAM (DPR) / PCI interface.

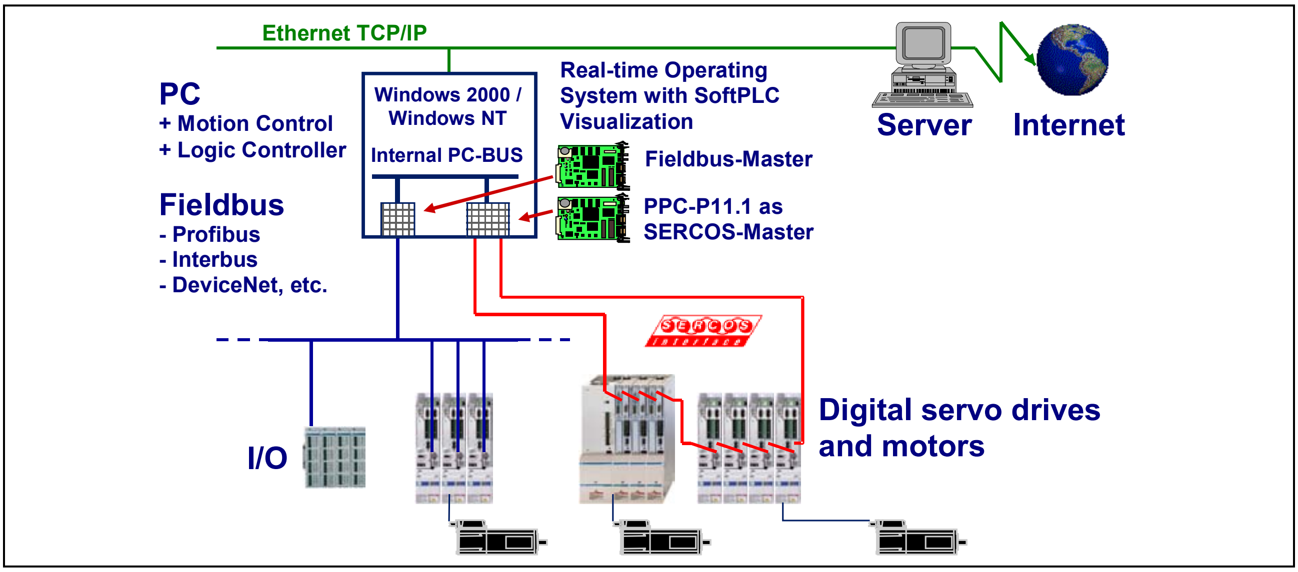


Fig. 20-1: PPC-P11.1 and PLC System

Fieldbus and I/O Support

The following table describes fieldbus and I/O supported by GMP 11 firmware:

Interface	Supported	Not Supported
Fieldbus Interface	Fieldbus master support when using the integrated PLC functionality (IndraLogic)	VisualMotion fieldbus slave interface cards
I/O Interface	Sercos Reco 02 modules, Sercos Inline modules , IndraDrive with MD1, ECO-X module and DIAX04 I/O cards	Local Reco 02 modules
Communication	Serial and PCI using VisualMotion Dde Server or SCP Server. Additional communication methods are available for both VisualMotion Toolkit and IndraLogic communication. see note 1	Direct EtherNet communication (The control can be access using Remote Access. see note 2)
Note 1: Refer to chapter 7, *IndraLogic Communication Channels*, in volume one for details. Note 2: Refer to chapter 7, *Remote Access to a VisualMotion Control*, in volume one for details.		

Table 20-1: Fieldbus and I/O Support

Rexroth Bosch Group

Rexroth Interfaces

The following PC interfaces are available from Bosch Rexroth.

Windows™ Environment

- VisualMotion Toolkit's Data Mapper (Fieldbus Mapper) for configuring the cyclic channel.
- VisualMotion DDE (VM_DDE) Server for communicating with VisualMotion Toolkit 10 via the DPR.
- Scalable Communication Platform (SCP) for communication with HMI products and VisualMotion Toolkit 10 via the DPR.

RTX (Real-time Operating System from VentureCom)

- Real-time Dynamic Linking Libraries (RTDLL) for interfacing with the Siemens WinAC PLC running in an RTX environment.

PCI Bus Memory

The PPC-P11.1 is identified on the PCI bus with the following vendor and device ID.

PCI	Value	Description
Vendor ID	0x16F2	Bosch Rexroth
Device ID	0x0001	PPC-P11.1
Base Address ID	0	used by PPC-P11.1 PCI memory

Table 20-2: PCI Identification

During power up, the PCI bus allocates a 32K memory block used by the following PPC-P11.1 memory structure. The listed address offset is added to the assigned PCI base address (0).

Memory	Address Offset	Length
Reserved	0x0000 – 0x03FF	1 KB
Operations Registers	0x0400 – 0x07FF	1 KB
Reserved	0x0800 – 0x3FFF	14 KB
Dual Port RAM	0x4000 – 0x5FFF	8 KB
Reserved	0x6000 – 0x7FFF	8 KB

Table 20-3: PCI Bus Address for the PPC-P11.1

20.2 DPR Interface

The DPR interface provides a common memory area accessible to system components for sharing and transmitting system data and status.

DPR Memory	Details
Register Channel	128 Input / 128 Output registers accessible to the PLC and PPC-P11.1
Cyclic Channel	64 Word Input / 64 Word Output, with optional multiplexing
Non-Cyclic Channel	Supports both modified Short Format 3 [Note 1.] (SF3a) and SIS [Note 2.]
Status and Control	Provides status and control of the SoftPLC and PPC-P11.1 Interface
PPC-P11.1 Diagnostic Information	Provides diagnostics and current Sercos phase
Operation Registers	Interrupt and mailbox registers used between the host (PC) and local (PPC-P-P11.1).
Programming Channel	Com channel used by Windows application through SCP or VM DDE Server
Note 1. Refer to the Fieldbus Interface chapters in the VisualMotion 11 Application Manual for Short Format 3 details. Note 2. Refer to chapter 12.4 for SIS details	

Table 20-4: DPR Memory Structure

The following figure illustrates the standard communication paths between the PC and the PPC-P11.1.

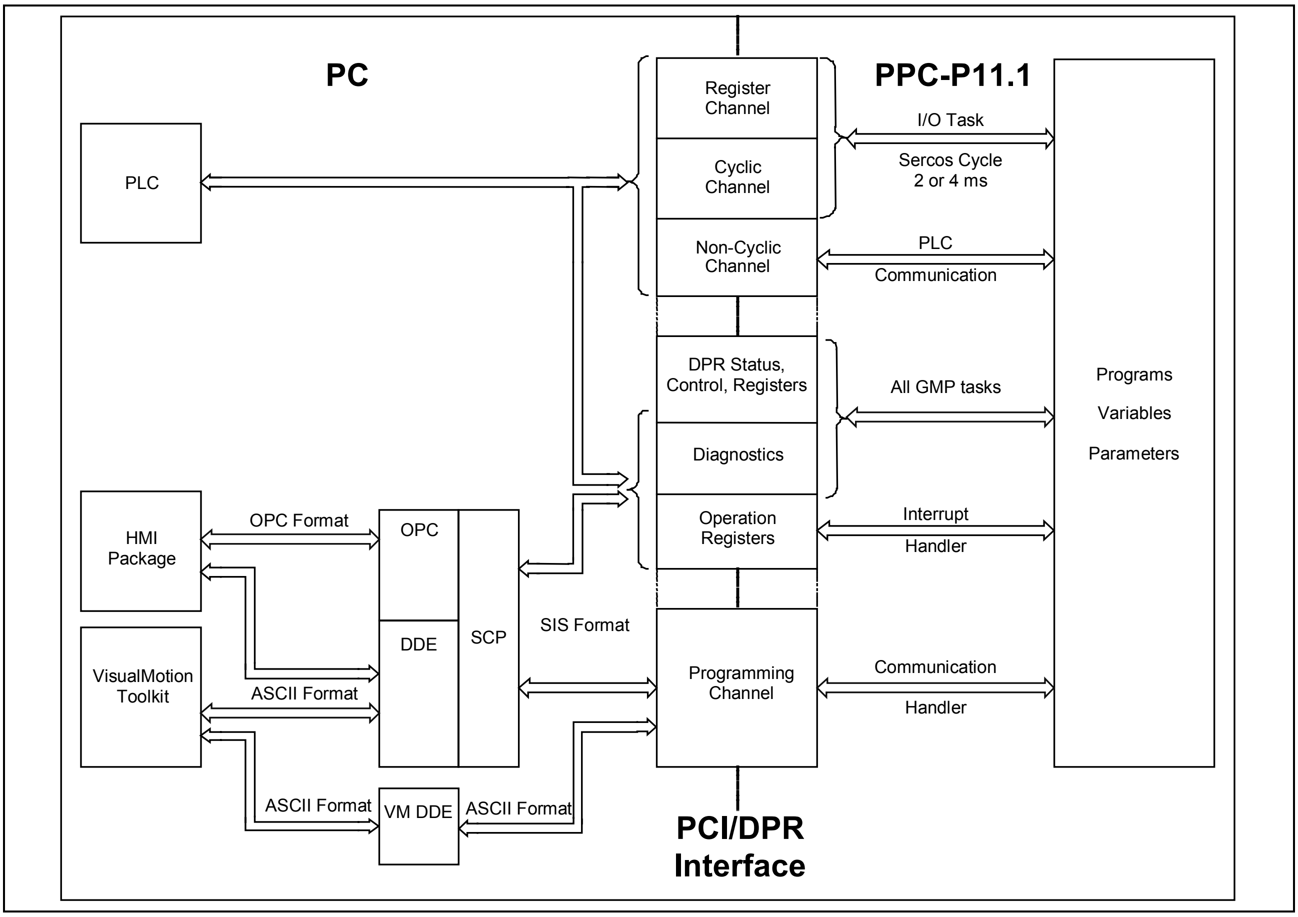


Fig. 20-2: Dual Port RAM Interface

Rexroth Bosch Group

DPR Parameters

The following tables list the parameters used to enable, configure and monitor the DPR interface between the PPC-P11.1 and the PC. Refer to chapter 5 for a detailed description of VisualMotion parameters.

PCI Communication

The following parameters enable communication across the PCI bus.

Parameter	Description
C-0-2635	Fieldbus/PLC Error Reaction
C-0-2640	Fieldbus/PLC Connection Options

Table 20-5: PCI Communication Parameters

Cyclic Channel

Parameter	Description
C-0-2600	PLC Mapper (cyclic channel) to PLC
C-0-2601	PLC Mapper (cyclic channel) from PLC
C-0-2632	PLC Multiplex Method
C-0-2636	PLC Word Swap
C-0-2638*	Fieldbus Available Cyclic IN Parameters
C-0-2639*	Fieldbus Available Cyclic OUT Parameters
***Note**: Read-only parameters list of valid parameters that can be used in C-0-2600/2601	

Table 20-6: Cyclic Channel Parameters

Register Channel

Parameter	Description
C-0-2641	PLC Input Register List
C-0-2642	PLC Output Register List

Table 20-7: Register Channel Parameters

PLC Monitoring

Parameter	Description
C-0-2637	PLC Firmware Version
C-0-2643	PLC Lifecounter Check: Number of Retries
C-0-2644	PLC Lifecounter Check: Current Number of Misses
C-0-2645	PLC Lifecounter Check: Peak Number of Misses
C-0-2646	PLC Lifecounter Check: Number of Timeouts

Table 20-8: PLC Monitoring Parameters

Cyclic Channel Monitoring

Parameter	Description
C-0-2611	PLC Cyclic Channel: Current number of misses
C-0-2612	PLC Cyclic Channel: Peak number of misses
C-0-2613	PLC Cyclic Channel: Timeout counter

Table 20-9: Cyclic Channel Monitoring Parameters

Register Channel Monitoring

Parameter	Description
C-0-2651	PLC Register Channel: Current number of misses
C-0-2652	PLC Register Channel: Peak number of misses
C-0-2653	PLC Register Channel: Timeout counter

Table 20-10: Register Channel Monitoring Parameters

Shared DPR Memory Map

The following table describes the shared DPR memory area for communication between a SoftPLC and the PPC-P11.1. All listed addresses are offset from the assigned PCI base address.

DPR Data		Address		Length	Direction		Comments
		Start (Hex)	End (Hex)	Bytes	PPC-P11.1	PLC	
PLC Cyclic Outputs	PPC Read Registers	0x4000	0x40FF	256	Read	Write	Defined by C-0-2642
	Cyclic Data Channel	0x4400	0x447F	128	Read	Write	Defined by C-0-2601; 64 words with multiplexing
PLC Cyclic Inputs	PPC Write Registers	0x4B00	0x4BFF	256	Write	Read	Defined by C-0-2641
	Cyclic Data Channel	0x4F00	0x4F7F	128	Write	Read	Defined by C-0-2600; 64 words with multiplexing
Non-Cyclic Channel	PLC Request	0x5800	0x590F	272	Read	Write	Modified SF3
	PPC Response	0x5A00	0x5B0F	272	Write	Read	or SIS
PPC-P11.1 Diagnostic Information	Error_Code	0x5E74	0x5E75	2	Write	Read	C-0-0123
	Diagnostic_ Text	0x5E7C	0x5EB7	60	Write	Read	C-0-0122 ; C-0-0124
	MC_Mode	0x5EB8	0x5EB9	2	Write	Read	C-0-0121 (Sercos phase)
Status, Control Handshaking	PLC_Stat	0x5F00	0x5F01	2	Read	Write	
	PLC_Cmd	0x5F02	0x5F03	2	Read	Write	
	PLC_Count	0x5F04	0x5F05	2	Read	Write	
	Reserved	0x5F06	0x5F09	4	Read	Write	
	PLC_Clock	0x5F0A	0x5F0D	4	Read	Write	writes to C-0-0126 when value changes PLC can set date/time on PPC-P11.1
	MC_Stat	0x5F20	0x5F21	2	Write	Read	
	MC_Response	0x5F22	0x5F23	2	Write	Read	
	MC_Count	0x5F24	0x5F25	2	Write	Read	
	Out_PLC	0x5F40	0x5F40	1	Read	Write	
	Out_MC	0x5F41	0x5F41	1	Write	Read	
	In_PLC	0x5F42	0x5F42	1	Read	Write	
	In_MC	0x5F43	0x5F43	1	Write	Read	
	INT_REQ_OS	0x5F4C	0x5F4C	1	No access	Write	
	INT_REQ_RTOS	0x5F4D	0x5F4D	1	No access	Write	
	INT_RES_OS	0x5F4E	0x5F4E	1	No access	Write	
	INT_RES_RTOS	0x5F4F	0x5F4F	1	No access	Write	
	PLC_Ident	0x5FA0	0x5FC7	40	Read	Write	
	MC_Ident	0x5FC8	0x5FEF	40	Write	Read	
	PLC_Phase	0x5FF0	0x5FF1	2	Read	Write	
	MC_Phase	0x5FF2	0x5FF3	2	Write	Read	
	PLC_Result	0x5FF4	0x5FF4	1	Read	Write	

Table 20-11: Shared DPR Memory

20.3 Register Channel

The register channel of the DPR is used to transmit VisualMotion register bits between the SoftPLC and the PPC-P11.1. GMP 11 firmware supports 1024 registers. Up to 128 registers can be written to each PLC Input (C-0-2641) and Output (C-0-2642) Register List. Each register is 16 bits (2 bytes) in length. Data consistency of all configured registers is transmitted to the DPR (I/O Task) every Sercos cycle (2 or 4 ms).

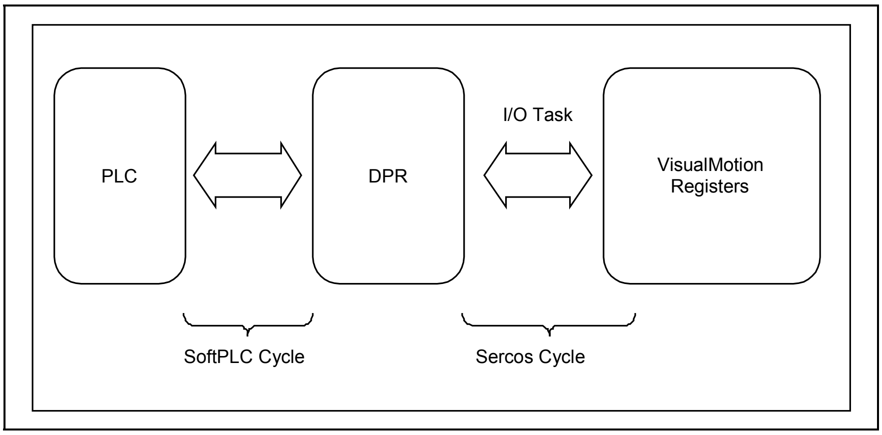


Fig. 20-3: Register Channel

Register Channel Transmission

The following table shows register transmission across the DPR.

Access		Direction	Length
PLC Input (Read)	PPC-P11.1 Output (Write)	PLC ← PPC-P11.1	256 Bytes
PLC Output (Write)	PPC-P11.1 Input (Read)	PLC → PPC-P11.1	256 Bytes

Table 20-12: PLC Input and Output Register List

Register Channel Configuration

The register channel is configured using VisualMotion Toolkit's Parameter Overview Tool. The registers configured in the Register Channel are transmitted in the order in which they appear in the PLC Input and Output Register Lists. Use the following steps to add registers:

1. Launch VisualMotion Toolkit in online or service mode, switch the control to parameter mode and select ***Data ⇒ Parameters…***
2. Double click on either register list (C-0-2641 or C-0-2642).
3. Right click in the *Edit* window and select ***Append Item (CTRL+INS)***
4. Add register numbers in the desired order of transmission.

Note: Read-only status registers must be added to the PLC Input Register List C-0-2641.

20.4 Cyclic Channel

The cyclic channel is used to transmit system parameters and program variables between the SoftPLC and PPC-P11.1. Data transfer is limited to 64 words, with optional multiplexing for up to 32 levels. Each parameter and variable is considered a 32-bit double word in length. Each register is 16 bits (2 bytes) in length. Data consistency of all configured parameters and variables is transmitted to the DPR every Sercos cycle (2 or 4 ms).

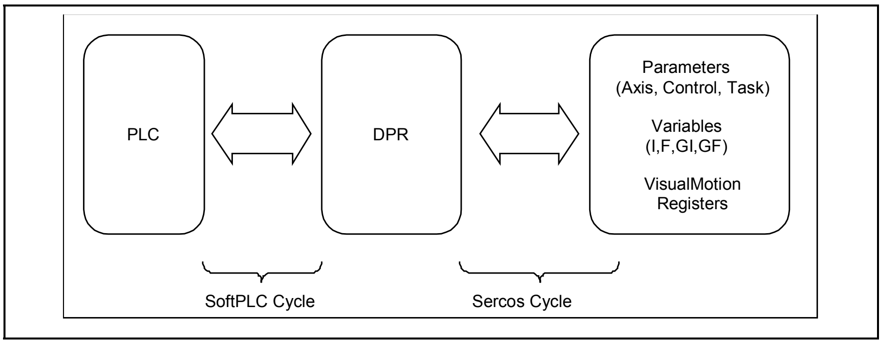


Fig. 20-4: Cyclic Channel Overview

Accessing Drive Parameters

Drive parameter (S and P) should be transmitted non-cyclically due to the inherent delay of parameter access over the Sercos service channel. However, if a drive parameter is mapped to an axis parameter, then the axis parameter can be transmitted as cyclic data. Control parameters C-0-2638 and C-0-2639 are lists of allowable parameters that can be mapped. Refer to axis parameters A-0-0180 through A-0-0196 in chapter 5 for details.

Cyclic Channel Transmission

The following table shows data access across the DPR.

Access		Direction	Max. Length
PLC Input (Read)	PPC-P11.1 Output (Write)	PLC ← PPC-P11.1	128 Bytes
PLC Output (Write)	PPC-P11.1 Input (Read)	PLC → PPC-P11.1	128 Bytes

Table 20-13: Cyclic Channel DPR

Cyclic Channel Configuration

The cyclic channel is configured using VisualMotion Toolkit's Data Mapper (Fieldbus Mapper). The parameters and variables configured in the cyclic channel are transmitted in the order in which they appear in C-0-2600 and C-0-2601. Refer to the fieldbus interface chapters in the VisualMotion 11 Application Manual for *Cyclic Data Configuration* details.

Note: When multiplexing, a maximum of 31 parameters and/or variables is allowed per level. Status and control words are 16 bits in length and are located at the end of configured data. Refer to the Profibus fieldbus interface chapter in the *VisualMotion 11 Application manual* for multiplexing details.

Handshaking

The DPR interface between the SoftPLC and the PPC-P11.1 verifies access to cyclic channel memory areas before reading or writing data (handshaking).

The SoftPLC sets its request register before checking to see if the PPC-P11.1 is using the memory area. If the DPR is busy, it keeps its request active until the memory area is available and then accesses the DPR.

The PPC-P11.1 will always check to see if the DPR is busy before setting its request bit for the DPR. If the SoftPLC is currently accessing the DPR, the PPC-P11.1 waits and tries once more in the same Sercos cycle. If the PPC-P11.1 is locked out, a missed cycle counter is incremented. After 10 consecutive missed cycles, an error or warning is issued based on the configured PLC Error Reaction parameter C-0-2635. The following control parameters are used to monitor handshaking conflicts.

Parameter	Description
C-0-2611	PLC Cyclic Channel: Current # of misses
C-0-2612	PLC Cyclic Channel: Peak # of misses
C-0-2613	PLC Cyclic Channel: Timeout Counter

Table 20-14: Handshaking Monitor Parameters

Note: If the error reaction is set to ignore errors, then no error or warning is issued, but the timeout counter (C-0-2613) is incremented.

Handshaking State Machine

The following tables contain the SoftPLC and PPC-P11.1 (MC) registers used for cyclic channel handshaking.

Name	Set By	Memory Area	Address Offset	Length
OUT_PLC	PLC	PLC Outputs	0x5F40	1 Byte
OUT_MC	PPC-P11.1		0x5F41	1 Byte
IN_PLC	PLC	PLC Inputs	0x5F42	1 Byte
IN_MC	PPC-P11.1		0x5F43	1 Byte

Table 20-15: Handshaking Registers

Handshaking Register Value	Description
0x00	Free area
0x80	Busy area

Table 20-16: Handshaking Register Value

The following figure flowcharts an example of the PLC Input Area.

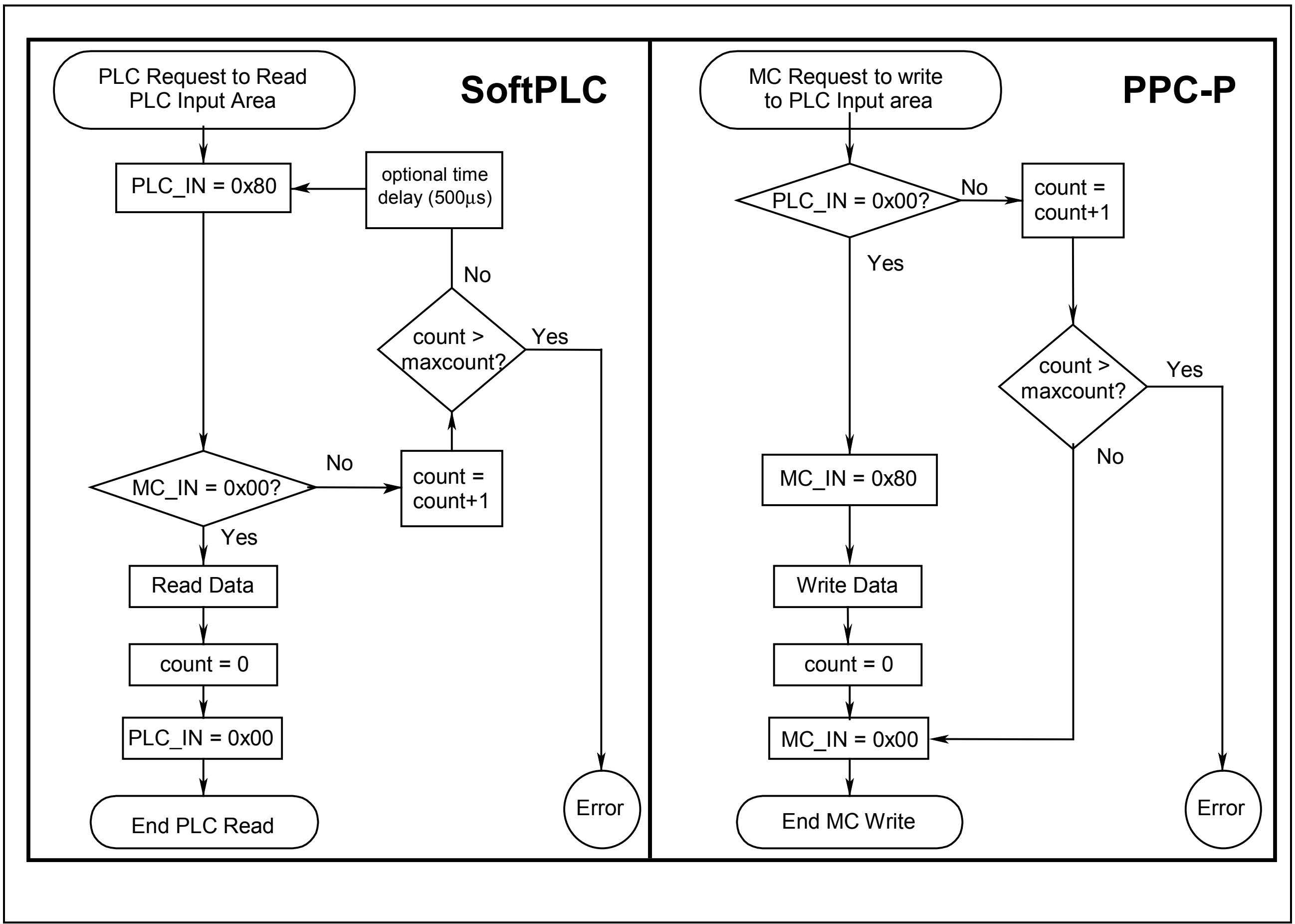


Fig. 20-5: Handshaking State Machine

The SoftPLC and PPC-P11.1 counters, in the above figure, are internal to each processor and increment every time the processor tries to access the DPR while in use. The counters are reset to 0 every time the read or write action is successfully completed.

Note: The SoftPLC should allow a time delay for the PPC-P11.1 to release the PLC Input memory area by setting IN_MC = 0x00, or set the loop *maxcount* to a large value.

20.5 Non-Cyclic Channel

The non-cyclic channel is used to transmit data that is not required every cycle, such as:

- parameter lists
- parameterization of axes or programs
- non-cyclically mapped data

List parameters must be transmitted non-cyclically due to their format. Drive parameter (S and P) must be transmitted non-cyclically due to the inherent delay of parameter access over the Sercos service channel.

The PPC-P11.1 is notified of a non-cyclic channel request with an interrupt and mailbox message from the SoftPLC. A response is returned from the PPC-P11.1 with another interrupt and mailbox message.

The non-cyclic channel supports the following two protocols:

Protocol	Description
SIS	standard Rexroth binary protocol using the Scalable Communication Platform (SCP)
Short Format 3 (SF3a)	modified version of Short Format 3

Table 20-17: Supported Protocols

Refer to the fieldbus interface chapters in the VisualMotion 11 Application Manual for Short Format 3 details. Refer to chapter 12.4 for SIS details

Protocol Identification

SIS allows access to all data. SF3a allows direct access to all non-list parameter values and can access lists, attributes, max/min values, etc., with ASCII protocol embedded in the data exchange objects. The first byte of the PLC Request identifies the protocol used.

Protocol	Value of First Byte
SIS	0x02
SF3a	0x20 - 0x5E

Table 20-18: Protocol Identification

Non-Cyclic Channel Transmission

The following table shows data access across the DPR.

DPR Data		Address	Direction	Length
PLC Request	(PPC-P11.1 Read)	0x5800	PLC → PPC-P11.1	272 Bytes
PPC Response	(PPC-P11.1 Write)	0x5A00	PLC ← PPC-P11.1	272 Bytes

Table 20-19: Non-Cyclic Channel DPR Location

Operating Procedure of the Non-Cyclic Channel

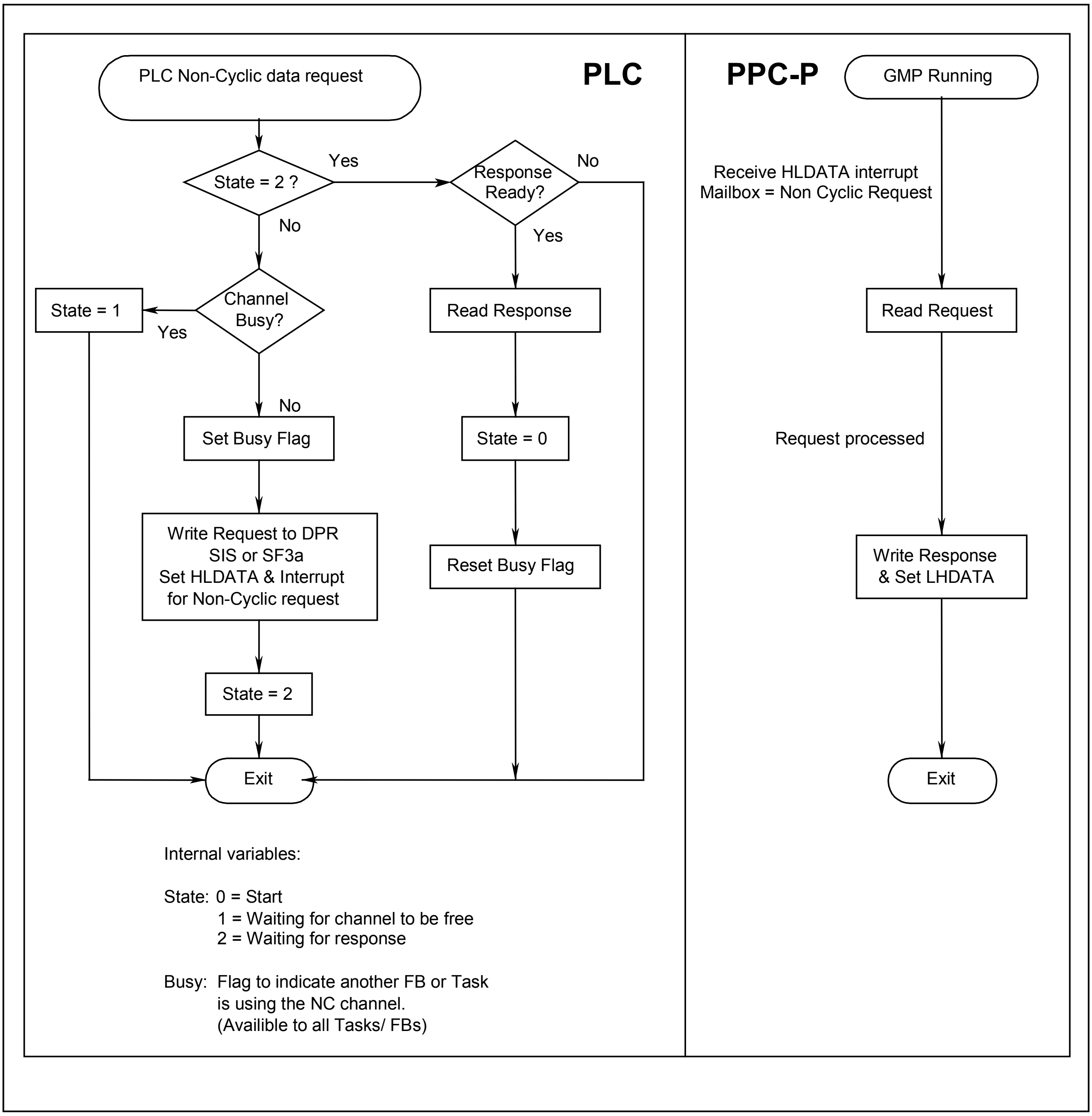


Fig. 20-6: Non-Cyclic Channel Operating Procedure

20.6 PPC-P11.1 Diagnostic Information

Error_Code

This parameter contains the current system status or error message number issued by the control.

Control parameter C-0-0123, Diagnostic Code is displayed.

If the SoftPLC interface is initialized, the update rate is based on the I/O task (2 or 4 ms).

Diagnostic_Text

Control parameters C-0-0122 Diagnostic Message, and C-0-0124 Extended Diagnostic Message are separated by a semicolon. The combined message is truncated to 60 bytes.

Diagnostic_Text is generated by the following rules:

1. If no extended message exists, C-0-0124 = 0, then Diagnostic_Text contains C-0-0122.
2. If an extended message exists, the first 30 characters in Diagnostic_text contain the first 30 characters of C-0-0122, followed by a semicolon (;), then followed by 29 characters of the extended message. In this case the message might not be complete.

For example: 412 No drives found on ring; CP0: Ring not closed

If the PLC interface is initialized, the update rate is based on the I/O task (2 or 4 ms).

MC_Mode

This parameter displays the current Sercos initializing phase of the control. The PPC-P11.1 mirrors C-0-0121 Current Sercos Phase in the DPR location.

If the SoftPLC interface is initialized, the update rate is based on the I/O task (2 or 4 ms).

20.7 Status and Control Registers

The following descriptions are for the Status and Control register of the DPR. Refer to Table 20-11: Shared DPR Memory for address offsets.

PLC_Stat

The following table describes the PLC status register.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Description	not used								Run	not used					Error	Ready

Table 20-20: PLC Status Register

Bit	Name	Description	Active	Actions
0	Ready	Indicates hardware/Firmware ready for startup	1	
1	Error	Indicates error on PLC system	1	Clear Ready and Running bits
7	Run	Indicates phase up complete; PLC Running	1	Start PLC program; Start life counter

Table 20-21: PLC Status Register Bit Description

PLC_Cmd

Following a request for PLC initialization, the PLC indicates which initialization sequence to startup. After a correct response is received from the PPC-P11.1 in MC_Response, this register is reset to 0x0000.

Value	Description
0x0000	No active command
0x0001	Request interface shutdown for new PLC program download
0x0002	Request MC startup initialization sequence

Table 20-22: PLC Command Register

PLC_Count

This 16-bit PLC life counter is checked by the PPC-P11.1 every Sercos cycle (In the I/O Task) to see if the PLC is active. This counter is written to on every PLC cycle.

PLC_Clock

The PLC_Clock register is written to by the PLC. The control reads this value and writes it to control parameter C-0-0126. The following table describes the PLC clock register.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Description	Hours					Minutes						Seconds in 2s increments				

Table 20-23: Low Word Bit Description

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Description	Day					Month				Year, relative to 1980						

Table 20-24: High Word Bit Description

MC_Stat

The following table describes the motion control status register.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Description	not used								Run	not used				/Download	Error	Ready

Table 20-25: Motion Control Status Register

Bit	Name	Description	Active	Actions
0	Ready	Indicates hardware/firmware ready for further startup	1	
1	Error	Indicates error on PPC-P11.1 (Mirror of PPC-P11.1 Error Bit	1	Clear Ready and Running bits
2	/Download	1 = Download not possible in Phase 3,4 0 = Allow PLC to download new programs when the PPC-P11.1 is in P2.	1 0	
6	pROBE	0 = PPC-P11.1 is operating normal 1 = PPC-P11.1 is in an exception state	0 1	 Reboot control
7	Run	Phase up complete; PPC-P11.1 is Running	1	Start life counter

Table 20-26: Motion Control Status Register Bit Description

MC_Response

During the handshaking sequence, the SoftPLC indicates which Handshaking sequence to start by writing 0x0001 or 0x0002 to the PLC_Cmd register after an interrupt and the initialization mailbox message.

The PPC-P11.1 acknowledges that it is following the correct Handshaking sequence by displaying the same value in MC_Response that the SoftPLC enters in PLC_Cmd.

Value	Description
0x0000	No active command
0x0001	Request interface shutdown for new PLC program download
0x0002	Request MC startup initialization sequence

Table 20-27: MC_Response

MC_Count

This 16 bit counter is incremented every Sercos cycle by the PPC-P11.1. The SoftPLC uses this counter as verification that the motion control is capable of communication. The counter starts after a successful initialization sequence by the SoftPLC.

Out_PLC

This Handshaking register is written to by the SoftPLC when requesting to write to the PLC_Output area (Both Register and Cyclic Data Channels). This register output is read by PPC-P11.1

Value	Description
0x00	Input/output area of DPR free
0x80	PLC accessing DPR

Table 20-28: PLC Output Handshaking Register

Out_MC

Handshaking register written by the PPC-P11.1 when requesting to read from the PLC_Output area (Both Register and Cyclic Data Channels). Read by SoftPLC.

Value	Description
0x00	Output area of DPR free
0x80	PPC-P11.1 accessing DPR

Table 20-29: Control Output Handshaking Register

In_PLC

Handshaking register written by the PLC when requesting to read from the PLC_Input area (Both Register and Cyclic Data Channels). Read by PPC-P11.1.

Value	Description
0x00	Input/output area of DPR free
0x80	PLC accessing DPR

Table 20-30: PLC Input Handshaking Register

In_MC

Handshaking register written by the PPC-P11.1 when requesting to write to the PLC_ Input area (Both Register and Cyclic Data Channels). Read by PLC.

Value	Description
0x00	Input area of DPR free
0x80	PPC-P11.1 accessing DPR

Table 20-31: Control Input Handshaking Register

INT_REQ_OS

Handshaking register written by the SCP when setting a PCI bus interrupt. (Read by the SoftPLC interface)

Value	Description
0x00	PCI host interrupt free
0x80	SCP accessing DPR

Table 20-32: Operating System Handshaking Request

INT_REQ_RTOS

Handshaking register, based on PLC interface, when requesting a PCI bus interrupt to the PPC-P11.1. (Read by the SCP)

Value	Description
0x00	PCI host interrupt free
0x80	PLC interface register / PCI host interrupt

Table 20-33: Real-time Operating System Handshaking Request

INT_RES_OS

Handshaking register written by the SCP when reading a PCI bus interrupt response from the PPC-P11.1. (Read by PLC interface)

Value	Description
0x00	Response free
0x80	SCP accessing DPR

Table 20-34: Operating System Handshaking Response

INT_RES_RTOS

Handshaking register, based on the PLC interface, when reading a PCI bus interrupt response from the PPC-P11.1. (Read by the SCP)

Value	Description
0x00	PCI response interrupt free
0x80	PLC interface request PPC-P11.1 response interrupt

Table 20-35: Real-time Operating System Handshaking Response

PLC_Ident

This string displays the customer-defined identification for the SoftPLC. This ID is read and displayed in control parameter C-0-2637.

MC_Ident

Optionally used during initialization by the SoftPLC, this string displays the installed firmware version in the PPC-P11.1.

For example: FWC-PFM01*-GP*-09V00

PLC_Phase

Indicates what phase the SoftPLC is in during the Start-up initialization sequence. The following table contains a brief overview of the PLC actions during these phases.

Value	PLC Action During this Phase
0x0000	Clear shared memory area in DPR (0x4000 – 0x5FFF) Set PLC_Stat ready (bit 0) high Set PLC start-up command (PLC_cmd = 0x0002)
0x0001	Check MC_Ident for valid string
0x0002	Set PLC_Stat Run (bit 7) high Start PLC program Start PLC_Count

Table 20-36: PLC Phase

MC_Phase

Indicates what phase the PPC-P11.1 is in during the Start-up initialization sequence. The following table contains a brief overview of the PPC-P11.1 actions during these phases.

Value	PPC-P11.1 Action During this Phase
0x0000	Write MC_Ident
0x0001	Set MC_Stat Ready (bit 0) high
0x0002	Set MC_Stat Run (bit 7) high Start MC_Count

Table 20-37: Control Phase

PLC_Result

(Optional)

SoftPLC writes 0x0001 after comparing MC_Ident to the valid firmware ID strings.

20.8 Operation Registers

The operation registers are used to set interrupts between the Host (PC) and the Local (PPC-P11.1) processor and for initialization of the PPC-P11.1/PLC interface, non-cyclic communication and the SCP interface. All addresses listed in the table below are offsets from the assigned PCI base address.

PCs using both a real-time operating system (RTOS) and a Windows based application should enable local to host mailbox interrupts using only the Windows interface. Sharing of the interrupt vector between two operating systems is not possible. The RTOS will poll the LHDATA mailbox for response to its interrupts set on the PPC-P11.1.

Operation Register	Address Offset		Length
	Start (Hex)	**End (Hex)**	**Bytes**
Host Interrupt Control/Status Register (**HINT**)	0x04E4	0x04E7	4
Host to Local Data Mailbox (**HLDATA**)	0x04E8	0x04EB	4
Local Processor Interrupt Control/Status Register (**LINT**)	0x04F4	0x04F7	4
Local to Host Data Mailbox (**LHDATA**)	0x04F8	0x04FB	4

Table 20-38: DPR Map

HINT - Host Interrupt Control and Status Register

The HINT operation register is used by the SCP and PPC-11.1 for setting a Local to Host Interrupt. This operation register works in conjunction with the LHDATA operation register mailbox. When a request for interrupt is set by the local (LHDATA, bit 24), bit 3 of the HINT interrupt status is enabled. The host then initiates the interrupt by enabling bit 19. After the interrupt is enabled, the contents of the LHDATA mailbox are read.

Note: All control and status bits are initially cleared on power-up.

Function	Address Offset	Bit	Description
Interrupt Event Status	0x04E4	3	0 = no events active (*default*) 1 = Local to Host mailbox
Interrupt Enable		19	0 = no interrupts are enabled (*default*) 1 = Local to Host mailbox interrupt enabled

Table 20-39: Host Interrupt Control and Status Register - HINT

LINT - Local Interrupt Control and Status Register

The LINT operation register is used by the PPC-P11.1 for setting a Host to Local Interrupt. This operation register works in conjunction with the HLDATA operation register mailbox. When a request for interrupt is set by the host (HLDATA, bit 24), bit 3 of the LINT interrupt status is enabled. The local then initiates the interrupt by enabling bit 19. After the interrupt is enabled, the contents of the HLDATA mailbox are read.

Note: All control and status bits are initially cleared on power-up.

Function	Address Offset	Bit	Description
Interrupt Event Status	0x04F4	3	0 = no events active (*default*) 1 = Host to Local mailbox
Interrupt Enable		19	0 = no interrupts are enabled (*default*) 1 = Host to Local mailbox interrupt enabled

Table 20-40: Host Interrupt Control and Status Register - HINT

Interrupt Mailboxes

The PPC-P11.1 uses the Host-to-Local and Local-to-Host Data Mailboxes in the Operation Registers to facilitate the interrupts sent from the PC (Host) to the PPC-P11.1 (Local) and in the other direction (Local to Host). Byte 0 and byte 1 contain the mailbox message read by the target processor to determine the reason for the interrupt.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Description	Mailbox High Byte								Mailbox Low Byte							

Table 20-41: Low Word Bit Description

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Description	not used							IRQ	not used							

Table 20-42: High Word Bit Description

LHDATA - Local to Host Data Mailbox

The LHDATA contains the requested local interrupt message. Bit 24 enables the local to send an interrupt to the host. When enabled, a mailbox interrupt is sent to the host processor. The interrupt remains active until cleared by the host in the HINT - Host Control and Status Register.

Function	Address Offset	Bit	Description
Data Word	0x04F8	7-0 (byte 0) 15-8 (byte 1)	Interrupt message written by the local and read by the host processor.
Interrupt to Host		24	0 = inactive 1 = active

Table 20-43: Host Interrupt Control and Status Register - HINT

HLDATA - Host to Local Data Mailbox

The HLDATA contains the requested host interrupt message. Bit 24 enables the host to send an interrupt to the local. When enabled, a mailbox interrupt is sent to the local processor. The interrupt remains active until cleared by the local in the LINT - Host Control and Status Register.

Function	Address Offset	Bit	Description
Data Word	0x04E8	7-0 (byte 0) 15-8 (byte 1)	Interrupt message written by the host and read by the local processor.
Interrupt to Local		24	0 = inactive 1 = active

Table 20-44: Host Interrupt Control and Status Register - HINT

Interrupt "Mailbox" Message Descriptions

Interrupt Requested	Value	Description
PLC initialization	0x0100	PLC has requested to initialize or stop communication with the PPC-P11.1 depending on whether PLC_Cmd = 0x0002 (initialize) or 0x0001(stop)
Non-Cyclic Message	0x0008	PLC requesting information over non-cyclic channel
Program Channel Message	0x0020	Request sent to PPC-P11.1 over Programming channel from PC user interface

Table 20-45: Interrupt Message Description

Clearing Interrupt Bits

Name	Address	Description
HINT	0x04E4	Host Interrupt Control/Status: Host to Local interrupt is cleared with a write to the Local Interrupt Control and Status Registers
LINT	0x04F4	Local Interrupt Control/Status: Local to Host interrupt is cleared with a write to the Host Interrupt Control and Status Registers

Table 20-46: Clearing Interrupt Bits

20.9 Interrupt Handshaking

PCI Interrupt to the PPC-P11.1

The flowchart below shows the procedure for setting a PCI interrupt to the PPC-P11.1.

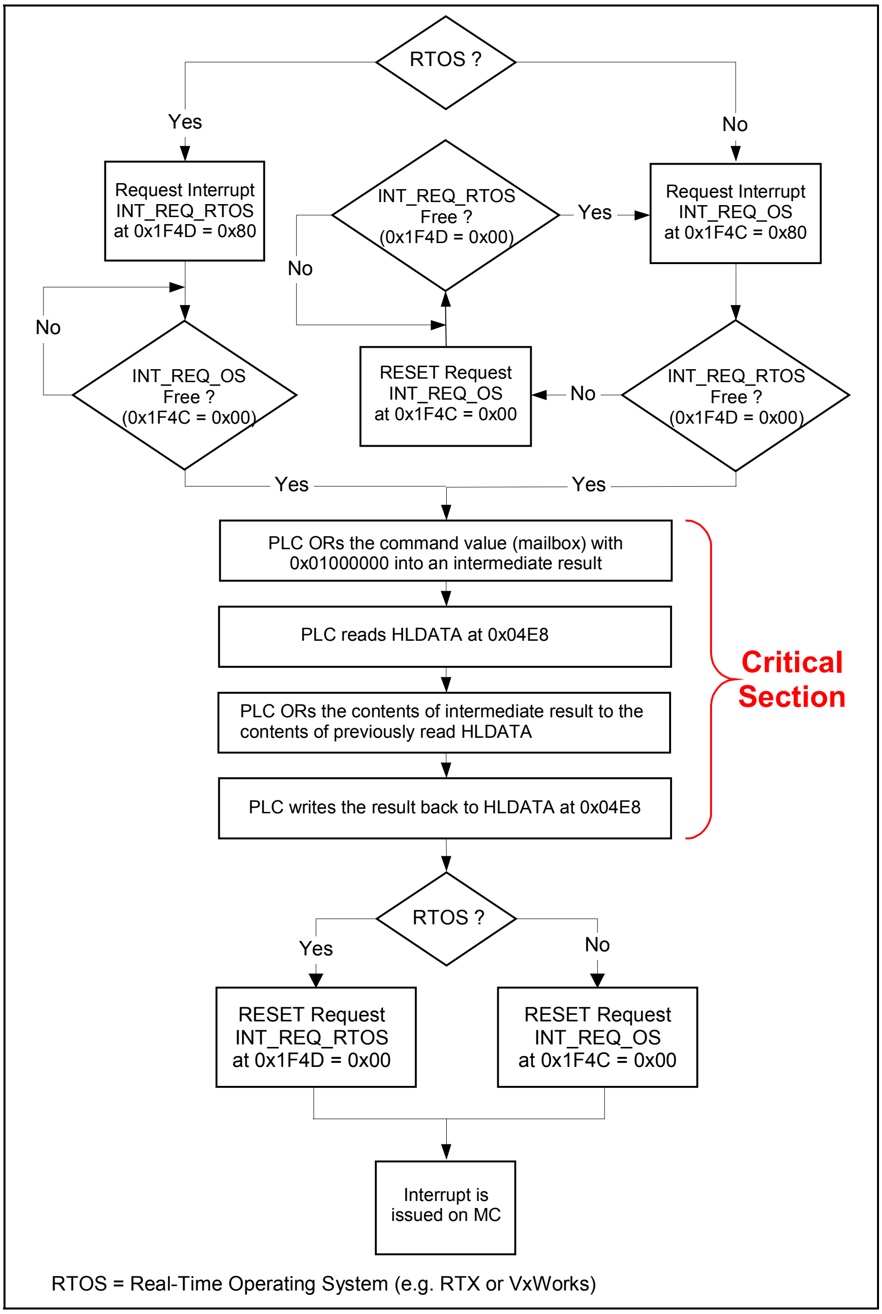


Fig. 20-7: PCI Interrupt to the PPC-P11.1

Receiving Interrupt Response from PPC-P11.1

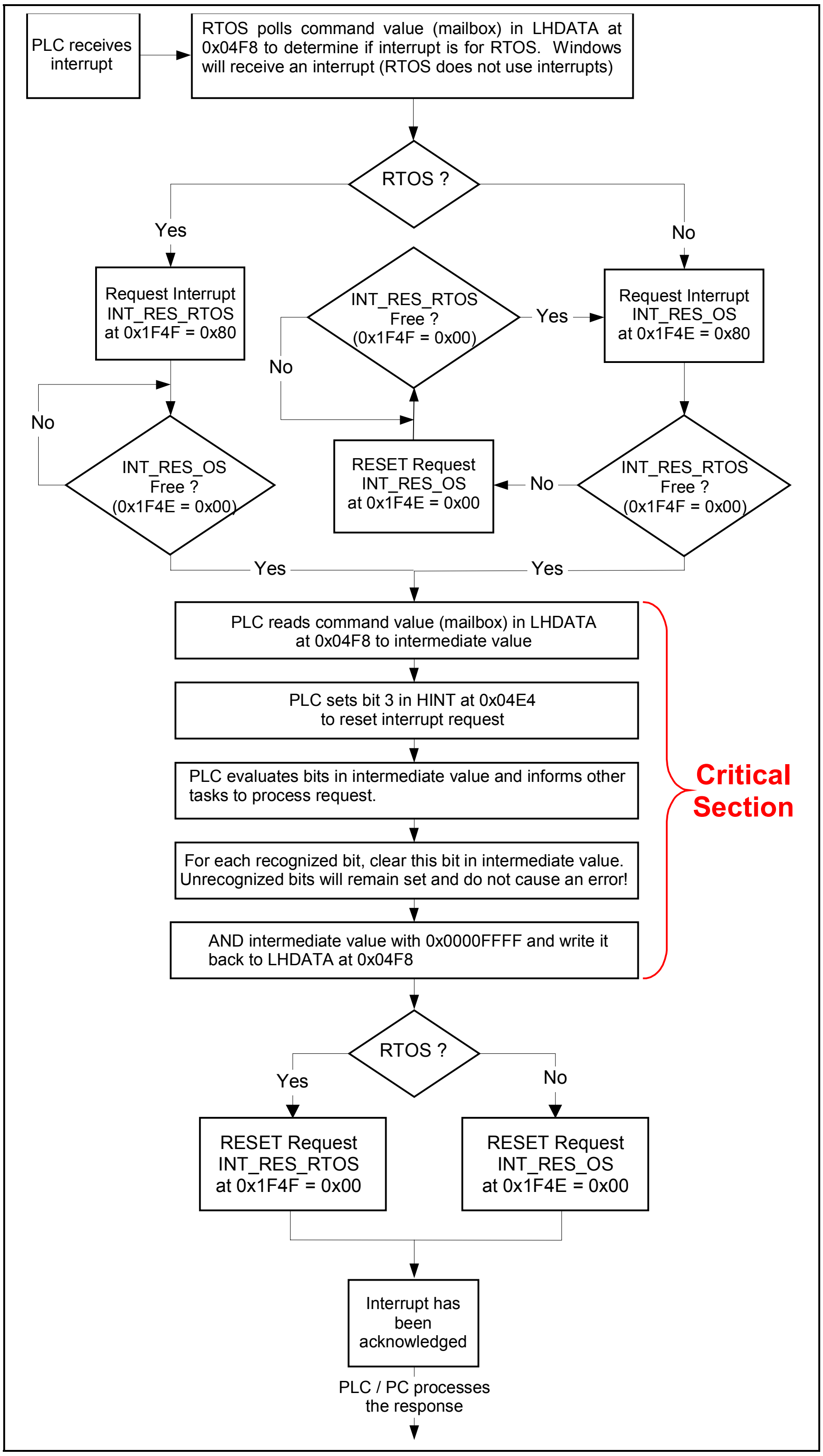


Fig. 20-8: Receiving Interrupt Response from PPC-P11.1

Enabling the PCI Interrupts

The enabling of PCI interrupts is performed by the Windows application (SCP or VM_DDE) and not by the real-time operating system (RTOS) application.

In the RTX application, only clear the handshaking registers and mailbox if the Windows application did not start. The mailbox enable in the HINT register is checked.

In the event that an application is running in a RTOS and the Windows based application has not started, the RTOS application should initialize the mailbox and handshaking registers.

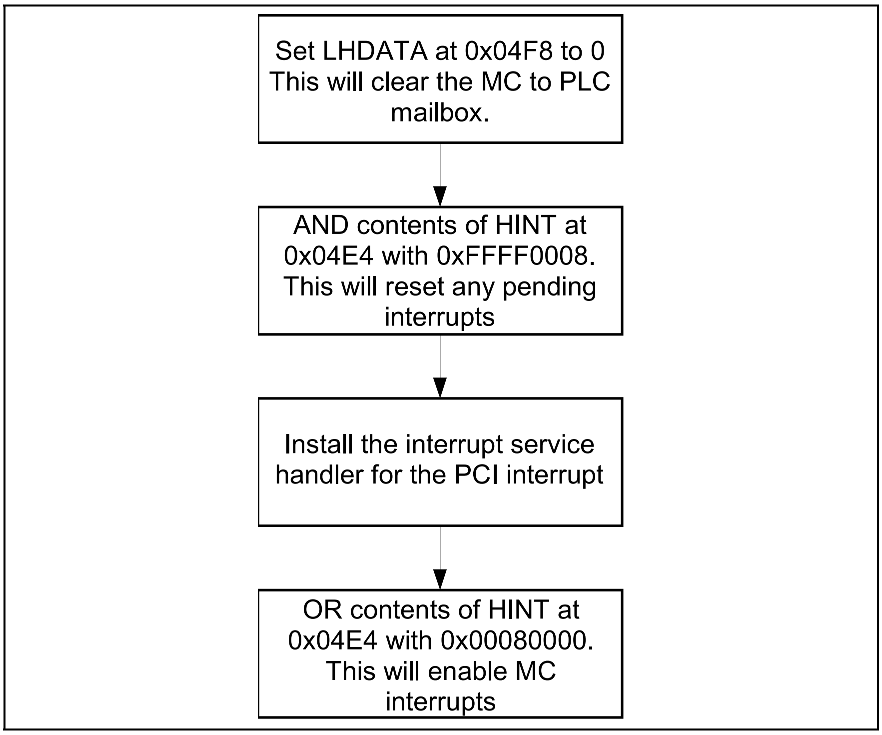


Fig. 20-9: Enabling the PCI Interrupts

20.10 System Initialization

PLC to PPC-P11.1 Communication

The following steps are used to initialize communication between a PLC and the PPC-P11.1:

1. Locate the PPC-P11.1 on the PCI Bus using the Vendor and Device IDs.
2. Map PPC-P11.1 Dual Port RAM areas to Virtual Memory on the PC. Most operating systems do not allow direct access from application programs to the actual hardware memory addresses. The operating system has standard functions in its API, which will handle this mapping. (RTX example: RtMapMemory)
3. Initialize communication between the SoftPLC and PPC-P11.1 using the handshaking method previously described.
4. The SoftPLC should monitor the PPC-P11.1's life counter while running its own life counter to maintain the connection. The easiest way to do this is to mirror back the PPC-P11.1's life counter as the SoftPLC life counter. As long as the lifecounters run, the data in the register, cyclic and non-cyclic channels will be valid. To maintain data integrity on the cyclic channel, use the handshaking registers.

PLC / PPC-P11.1 Initialization Sequence

PLC Phase	Control Phase	Time	SoftPLC	Control (MC)
0	0	0	[PLC starts] PLC checks and clears DPR (0x4000...0x5FFF) PLC sets PLC_Stat, Ready = 1 (Bit 0) PLC sets PLC_Phase = 1 "Start Up Command to MC (PLC_CMD = 0x0002) PLC sets IRQ, Mailbox value: 0x0100	Internal tests MC Resets IRQ; Waits for PLC_Phase = 1 [Indicates Waiting For PLC if PLC communication option is set after initialization]
1	0	1	Waits for MC_Phase = 1 Timeout: 20s	MC copies "FWC-PFM01*-GP*-09VRS-MS" into MC_Ident MC sets MC_Phase = 1 MC sets MC_Stat Ready =1 (Bit 0)
1	1	2	(Optional) PLC checks for MC_id If OK: PLC sets PLC_Phase = 2 Else Incorrect firmware error If do not check ID: PLC sets PLC_Phase = 2	Waits for PLC_Phase = 2
2	1	3	Waits for MC_Stat, Ready = 1 (Bit 0) Timeout 1s	MC sets MC_Stat Ready = 1 (Bit 0) MC sets MC_Phase = 2 MC sets MC_Stat, Run = 1 (Bit 7) Start Life Counter
2	2	4	PLC starts PLC program PLC set PLC_Stat Run =1 (Bit 7) Start Life Counter Normal running	Normal running (Life Counter Active) If Sercos Phase 2 or Lower: Set MC_Stat, /Download = 0 (Bit 1) Else Set MC_Stat, /Download = 1(Bit 1)

Table 20-47: PLC / PPC-P11.1 Initialization Sequence

PLC / PPC-P11.1 Program Download or Shutdown Sequence

When new programs are activated on the SoftPLC, the life counter may not be updated in the required length of time causing a VisualMotion error. The life counter is ignored when the SoftPLC indicates a program download or shutdown activation.

Note: Following a shutdown, the PPC-P11.1 is not allowed to phase up to phase 4 until a connection is re-intialized or PLC_CMD is not equal to 0x01. MC_Count is not updated.

PLC Phase	Control Phase	Time	SoftPLC	Control (MC)
2	2	0	Normal running	Normal running
2	2	1	If MC_Stat Bit 2 "/Download" = 0 Clears PLC_Stat Run (Bit 7) “Shut Down Command” to MC (PLC_CMD=0x0001) PLC sets IRQ, Mailbox value: 0x0100 Else Error message “Download not possible” ***Remedy***: Switch control in and out of parameter mode and try again.	Normal running: PLC_CMD = 0x0001 indicates program download, ignore PLC life counter if IRQ also received. Note: MC_Stat bit 2 / Download (1 = Sercos phase 3, 4. 0 = Sercos phase 2 or lower)
2	2	2		MC leaves run-mode if necessary
0	0	3	re-initialize connection	

Table 20-48: PLC / PPC-P11.1 Program Download or Shutdown Sequence

20.11 Designing a Custom Interface

The PPC-P11.1 has the following 3 common levels of interface and 10 common levels of functionality:

Interface	Description	Functionality
Hardware	Locating the PPC-P11.1 and configuring the PCI interface.	• Finding the PPC-P11.1 • Mapping physical memory to PC virtual memory • Un-mapping virtual memory when application is terminated
Communication	Initialize and maintain communication	• Initializing connection between PLC and PPC-P11.1 • Running and monitoring lifecounters • Monitoring Diagnostics • Shutting down the connection between PLC and PPC-P11.1 for short periods of time
Data Exchange	Transfers data between SoftPLC and VisualMotion system	• Exchanging cyclic channel data • Exchanging register channel data • Non-cyclic channel communication

Table 20-49: PLC and PPC-P11.1 Common Interface and Functionality

SoftPLC to PPC-P11.1 Communication

The following steps are used for communication between a PC based PLC (SoftPLC and Slot PLC) and the PPC-P11.1:

1. Locate the PPC-P11.1 on the PCI Bus using the Vendor and Device IDs.
2. Map PPC-P11.1 Dual Port RAM areas to Virtual Memory on the PC. Most operating systems do not allow direct access from application programs to the actual hardware memory addresses. The operating system has standard functions in its API, which will handle this mapping. (RTX example: RtMapMemory())
3. Initialize communication between the SoftPLC and PPC-P11.1.
4. The SoftPLC should monitor the PPC-P11.1's life counter while running its own life counter to maintain the connection.
5. Copy the initialization status and control registers and the PPC diagnostics to the RTOS application for viewing by user.
6. Exchange data, cyclic and non cyclic, as long as the connection is initialized and the lifecounters are running. To maintain data integrity on the cyclic channel, use the handshaking registers.
7. (Optional) The lifecounter monitoring can be shut off for a short period of time, for example during a program download to a soft PLC. In this case, the PPC-P must be in Sercos phase 2 and will not be allowed back to Sercos phase 4 until the connection is re-initialized.
8. When the connection is ended, the virtual memory used to map the PPC-P's DPR must be un-mapped.

Locating the PPC-P and Configuring the PCI Interface

Some controls provide a generic PCI driver that will connect to any PCI card given the vendor ID, Device ID and base address, while others provide a framework, but the programmer must configure everything.

For the control that does not use a generic driver, the first step is to find the PPC-P on the PCI bus by searching each slot in every PCI bus on the PC. Once found, store the card information so that the card's DPR can be mapped to virtual memory.

Note:	Currently only 1 PPC-P card installed in a PC is supported.

If the PPC-P card is not found:

- Verify the PPC-P card is installed in the PC
- Check the PPC-P 7 segment display. If only a single dot is visible or the display is blank, verify that the PC is supplying the required current from the 3.3V power supply.
- Use a PCI utility program, there are many freeware versions available, to see if the PPC-P card is visible on the PCI bus.

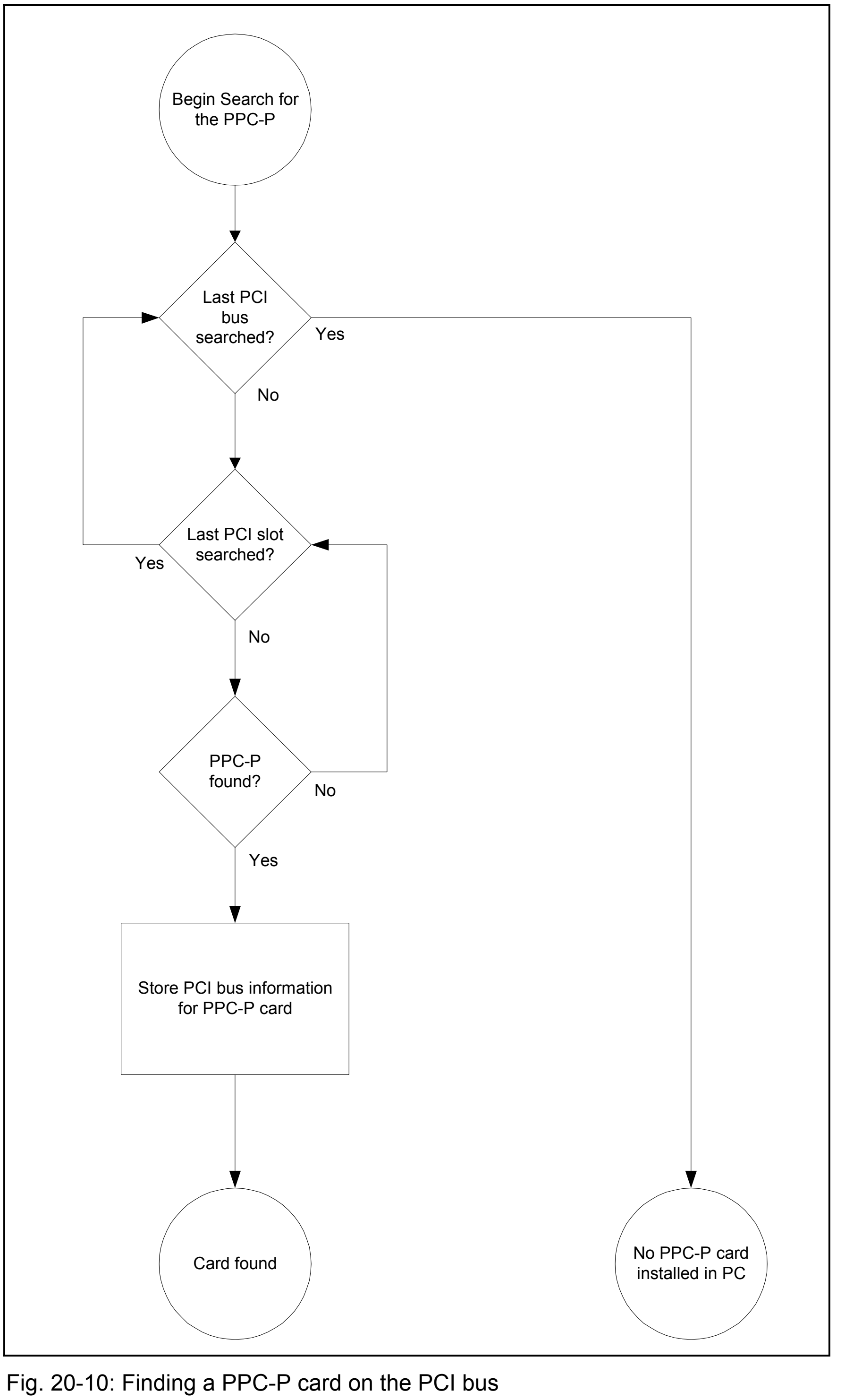


Fig. 20-10: Finding a PPC-P card on the PCI bus

Mapping the DPR to Virtual Memory

Most operating systems do not allow the user to access the physical memory of the PC or PCI cards directly, but instead force the user to map the memory to virtual memory on the PC.

Table 20-50 shows the definition of the full 32K defined from the Base Address [0] offset. As shown in the table, there are many areas that are either reserved or undefined, which should not be written to in the custom application. It is possible to map the full memory area from 0x00 to 0x5FFF, but care must be taken to not to write to any of the reserved memory locations.

The recommended method is to map each section individually, which would also shrink the amount of memory used by the application. One possible mapping is shown in the next table

Memory Area	Start Address	Bytes Mapped
Operation Registers	0x0400	256
PLC Cyclic Outputs	0x4000	1104
PLC Cyclic Inputs	0x4B00	1104
Non-cyclic Channel	0x5800	784
PPC-P Diagnostics	0x5E70	70
Status/Control/Handshaking	0x5F00	256

Table 20-50: DPR Memory Map

Note: RTX does not allow addresses other than ones ending with 0 to be used with the RtTranslateBusAddress() and RtMapMemory(). For example, if a memory address of 0x5E74 was passed into RtTranslateBusAddress(), the function will not return an error, but instead return a pointer to the address of 0x5E70 that will be used with RTMapMemory().

Un-Mapping Virtual Memory

To avoid memory leaks, the virtual memory where the PPC-P was mapped must be un-mapped when the application is shut down.

Initializing Connection Between SoftPLC and PPC-P

Once the DPR has been mapped to DPR, the connection between the SoftPLC and PPC-P can be initialized. This can be done in many ways, but in this case, a new thread has been created in RTX to call the example function so that the SoftPLC cycle is not extended while waiting for the response states from the PPC-P.

For the optional GMP firmware typecode check, only the main release needs to be checked and not the complete version. So only a subset of MC_Id would need to be checked. Example: PFM01*-GMP-10

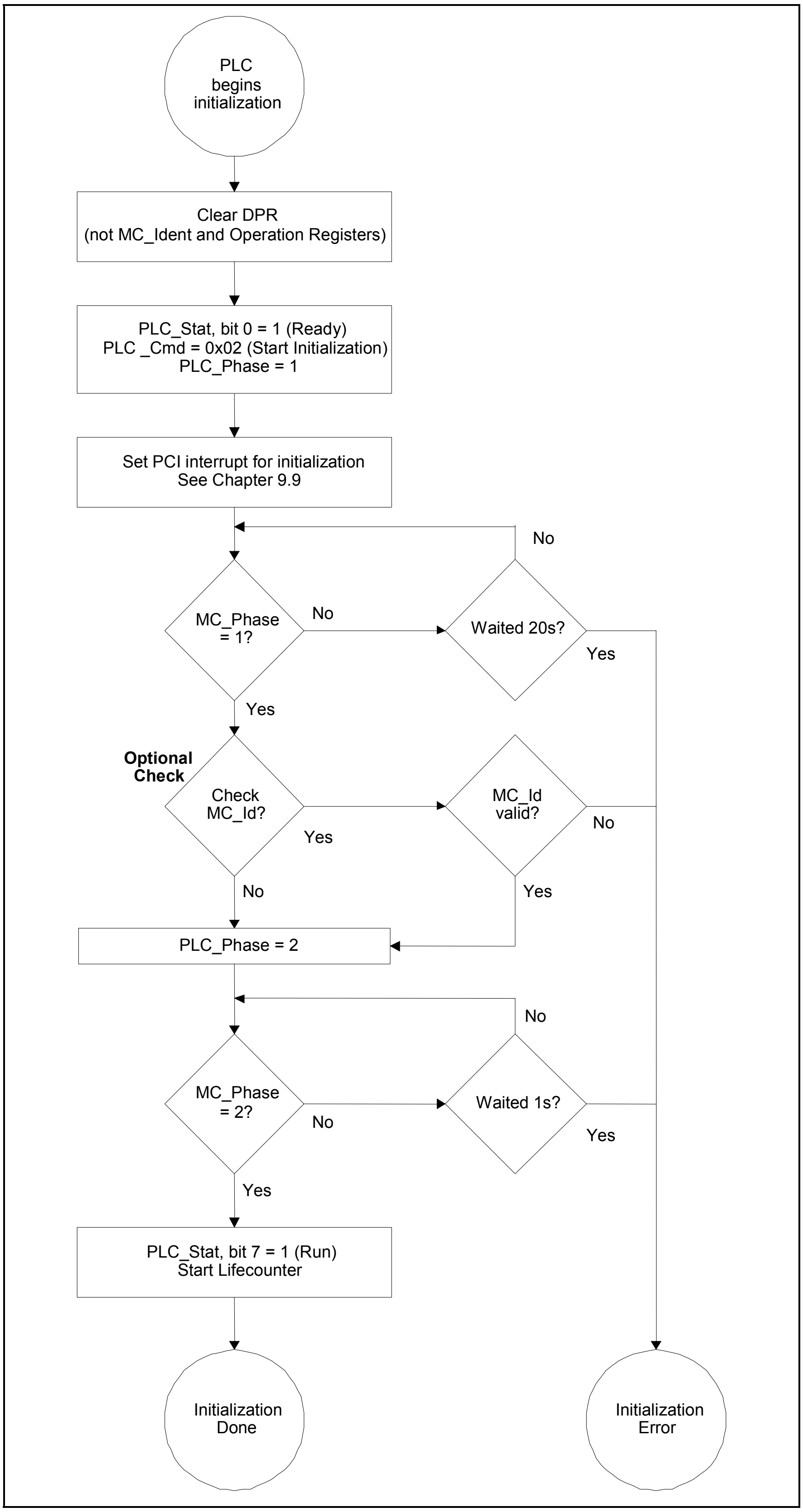


Fig. 20-11: Initializing the PPC-P Interface

Running and Monitoring Lifecounters

The SoftPLC should monitor the PPC-P11.1's life counter while running its own life counter to maintain the connection. The easiest way to do this is to mirror back the PPC-P11.1's life counter as the SoftPLC life counter. As long as the lifecounters run, the data in the register, cyclic and non-cyclic channels will be valid.

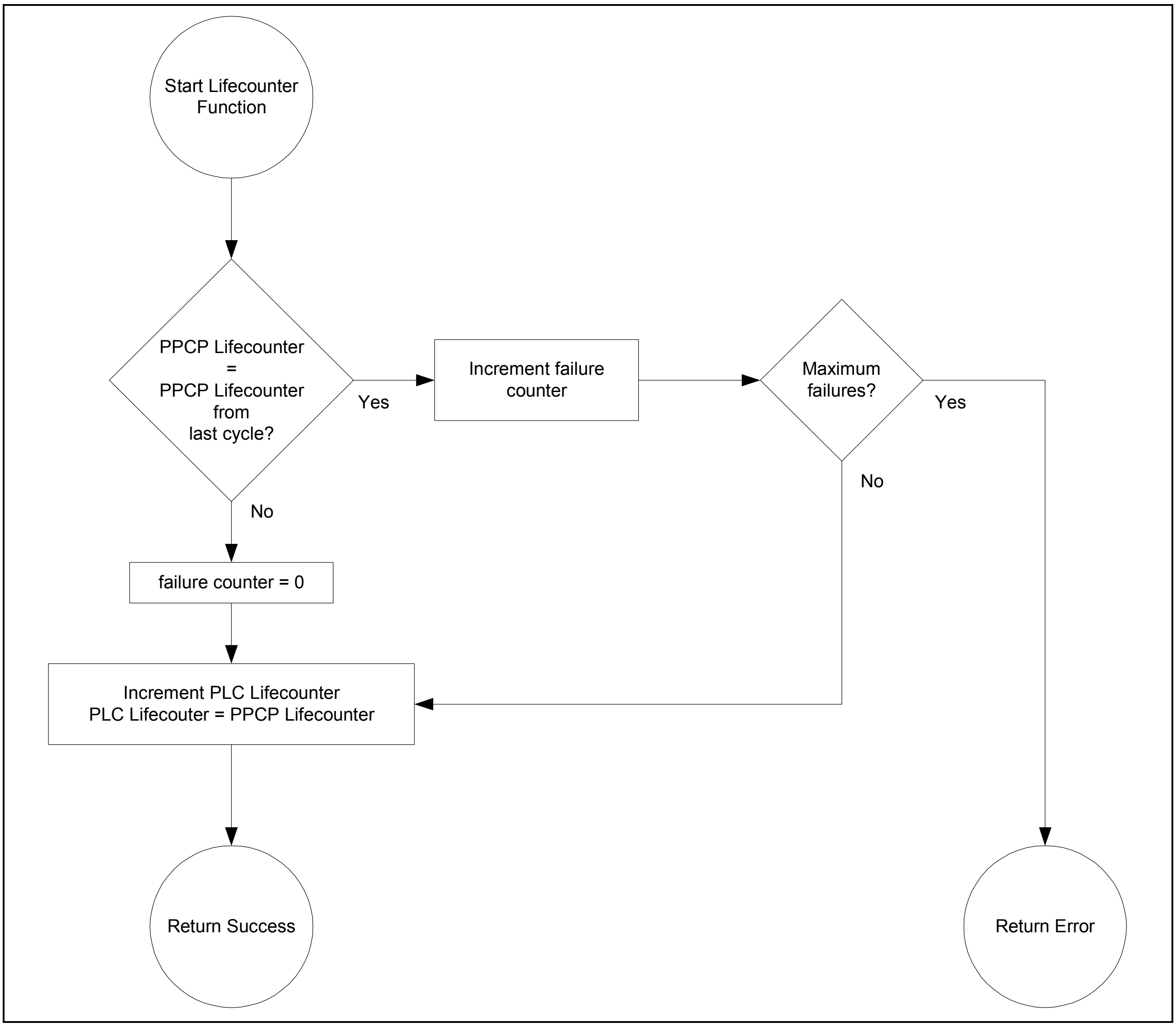


Fig. 20-12: Lifecounter Monitoring

Monitoring Diagnostics

The PPC-P diagnostic control parameters C-0-0121 to C-0-0124 are mapped directly to the Dual Port RAM and are available in the following locations

- Error_Code = C-0-0123
- Diagnostic_Text = C-0-0122; C-0-0124
- MC_Mode = C-0-0121

When the interface is initialized, the values are updated with the I/O task every 2 or 4 ms.

Shutting Down the Connection Between SoftPLC and PPC-P

This optional procedure should only be used for cases where the PPC-P should stop its lifecounter monitoring for short periods of time, for example downloading and activation of a new SoftPLC program.

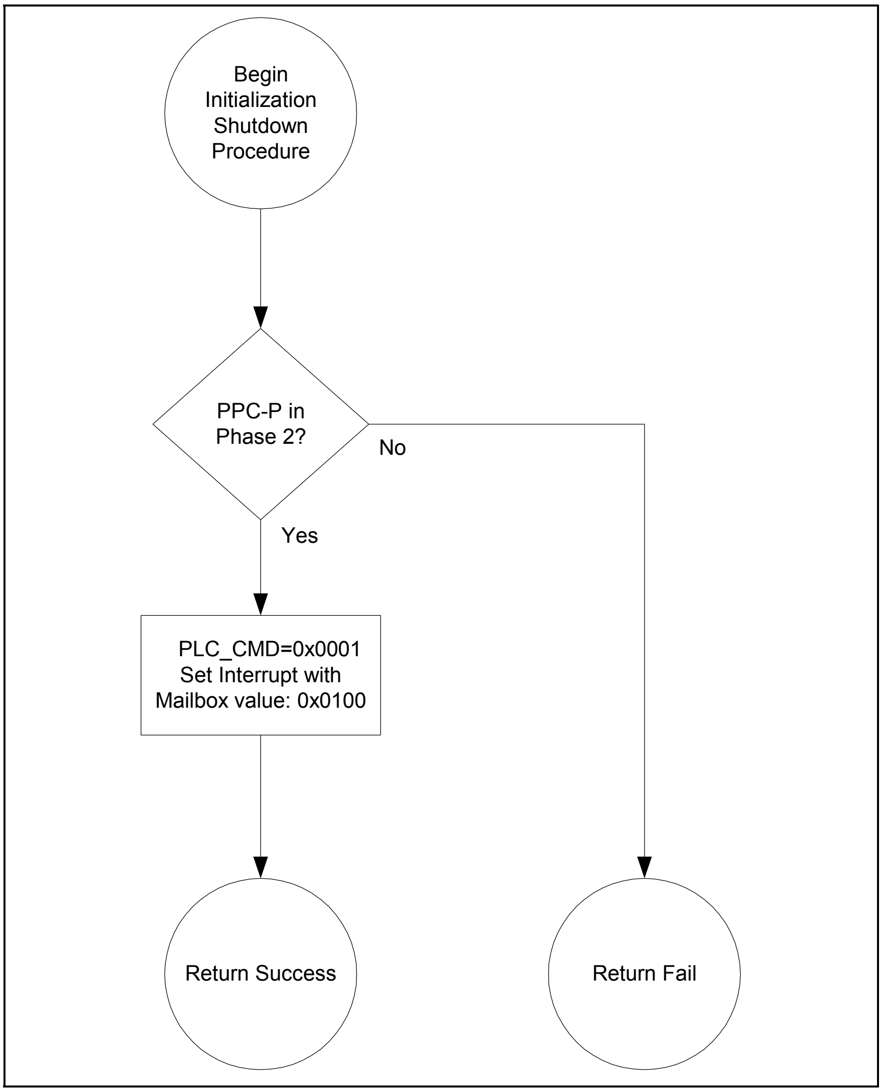


Fig. 20-13: Initialization shutdown

Note: The PPC-P must be in Sercos phase 2 and will not allow the user to phase up to Sercos phase 4 until the connection is re-initialized.

Exchanging Cyclic Channel Data

When the connection between the PC based control and the PPC-P is initialized and the lifecounters are running, it is possible to exchange data. To guarantee data consistency over the whole cyclic channel in each direction, the handshaking registers should be used. There are many methods to handle the handshaking and monitor how often the data cannot be transferred due to the PPC-P accessing the data at the same time. In the example illustrated in Fig. 20-14, the function will return either success that both inputs and outputs were transferred, or which channel failed. The PLC programmer is then responsible to retry and keep track of the failures.

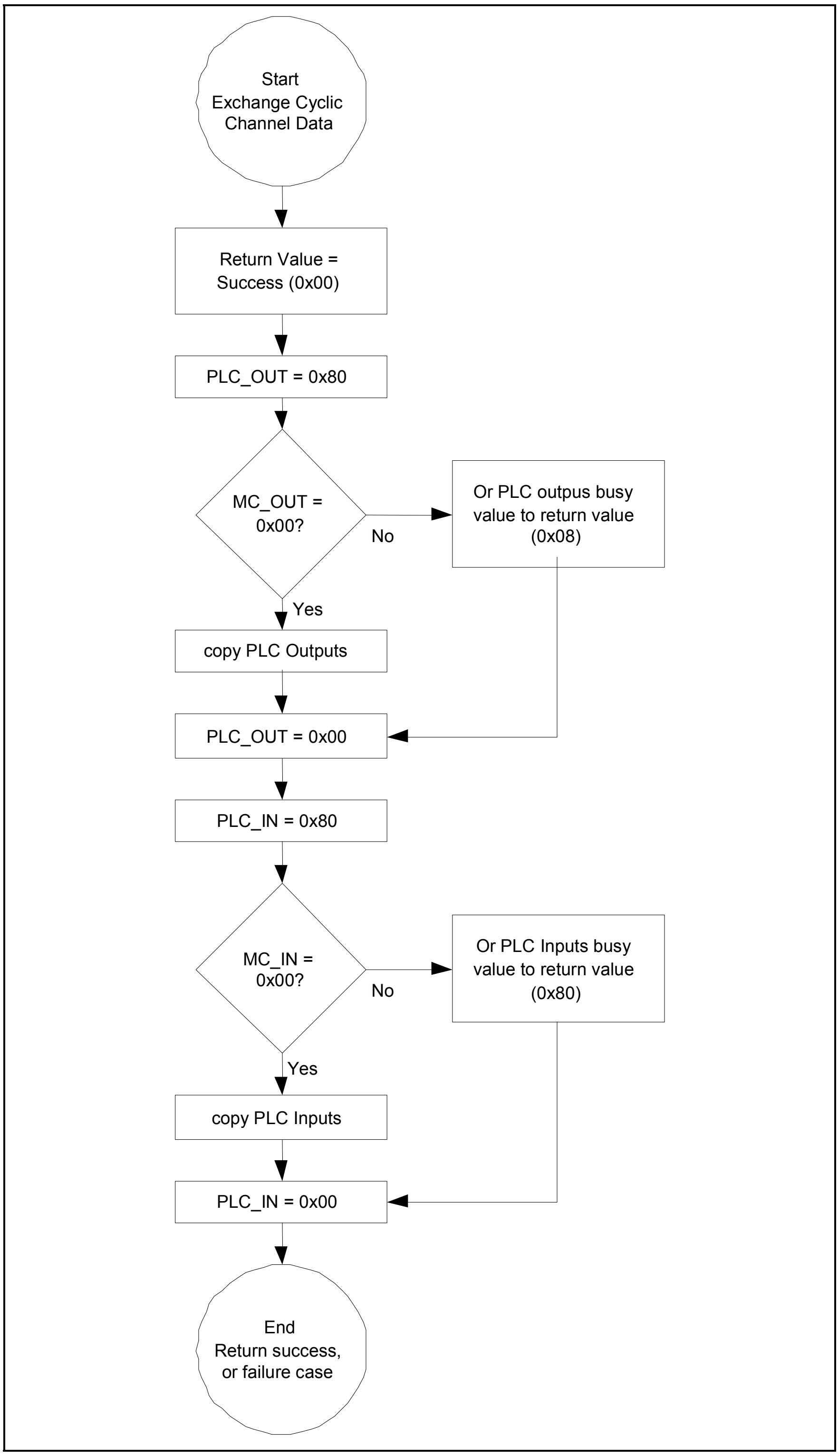


Fig. 20-14: Cyclic channel data transfer example

Suggestions

- GMP firmware attempts to copy the cyclic channel twice a cycle if the handshaking fails on the first attempt, and will issue an error if it is not able to copy the data for 10 cycles in a row. When asserting, PLC_IN and PLC_OUT, care must be taken so that they do not stay set for more than 20 or 40 ms, or else the PPC will error out with either a Cyclic timeout warning or error depending on how the error reaction is set in C-0-2635.
- If multiplexing or individual registers will be used, then data transfer should be done as individual words instead of double words. It is

possible that some controls will perform the conversion of data from Motorola to Intel. In this case, it may be possible that the individual words will not be in the expected location to be read from the Dual-Port RAM. See the cyclic channel example in the next section for the interface to the Siemens WinAC RTX plc.

- When the PLC fails to copy to the DPR, it may be desired to possibly retry the failed copy after a short wait, or after running some more functions to see it the PLC has freed the DPR.

Exchanging Register Channel Data

The register channel does not have any handshaking associated with it. Both controls can potentially access the same channel at the same time, so data consistency can not be guaranteed over the whole channel, but the PCI bus controller will guarantee that each individual register will be consistent.

Non-cyclic Channel Communication

Although both the Rexroth standard SIS protocol and Short format 3a are supported, the following example will focus on Short Format 3a. SF3a is a slightly modified version of the Short Format 3 protocol described for the fieldbus communications. The difference is that the length of the data transferred to or from the PPC-P is indicated in the DPR. The formats of these channels are shown in the next tables:

Byte	0x5800	0x5801	0x5802	0x5803	0x5804	0x5805	0x5806 …
Description	Index		Sub Index		Length	Reserved	Data (255 bytes maximum)

Table 20-51: PLC Request format of Non-Cyclic channel

Byte	0x5A00	0x5A01	0x5A02 …
Description	Error	Length	Data (255 bytes maximum)

Table 20-52: PPC-P Response format of Non-Cyclic channel

To ensure that the data is written and read from the correct byte location on the DPR, it is suggested that the PLC Request and PPC-P response channels are treated as arrays of bytes. Examples are shown in the next tables:

0x5800	**0x5801**	**0x5802**	**0x5803**	**0x5804**	**0x5805**	**0x5806 …**
0x48	0x02	0x00	0x01	0x00		

Table 20-53: Non cyclic request to read a mapped object (C-0-0002)

0x5800	**0x5801**	**0x5802**	**0x5803**	**0x5804**	**0x5805**	**0x5806 …**	**0x5807**
0x5A	0xEE	0x00	0x64	0x02		0x12	0x34

Table 20-54: Non cyclic request to write 0x1234 to a mapped object (Reg 100)

0x5800	**0x5801**	**0x5802**	**0x5803**	**0x5804**	**0x5805**	**0x5806 … 0x5812**
0x5E	0x74	0x00	0x00	0x0D		>1 DP 1.95 \r\n

Table 20-55: Non cyclic request using the data exchange object (S-0-0095)

Error Notification

For direct mapped objects, the errors will be indicated in the first byte of the PPC-P response channel (0x5800), and will be either one of the codes listed in Table 9-55 or one of the VisualMotion errors listed in the Communication Error Codes and Messages section of the manual.

With the data exchange objects, only the error codes listed in table 9-55 will be indicated in the first byte of the PPC-P response channel.

Error	Description
0xF0	ASCII Error: Illegal ASCII string
0xF2	Index Error: Invalid Index
0xF3	Sub-Index Error: Invalid Sub-Index

Table 20-56: Data Exchange Object errors

Otherwise, the errors will be indicated in the first 3 bytes of the PPC-P response data, where the error is preceded by and exclamation point (!).

The programmer is allowed to define other error codes to customize their interface as long as the number is not already defined by VisualMotion.

Note: Only 1 non-cyclic request is allowed at a time!

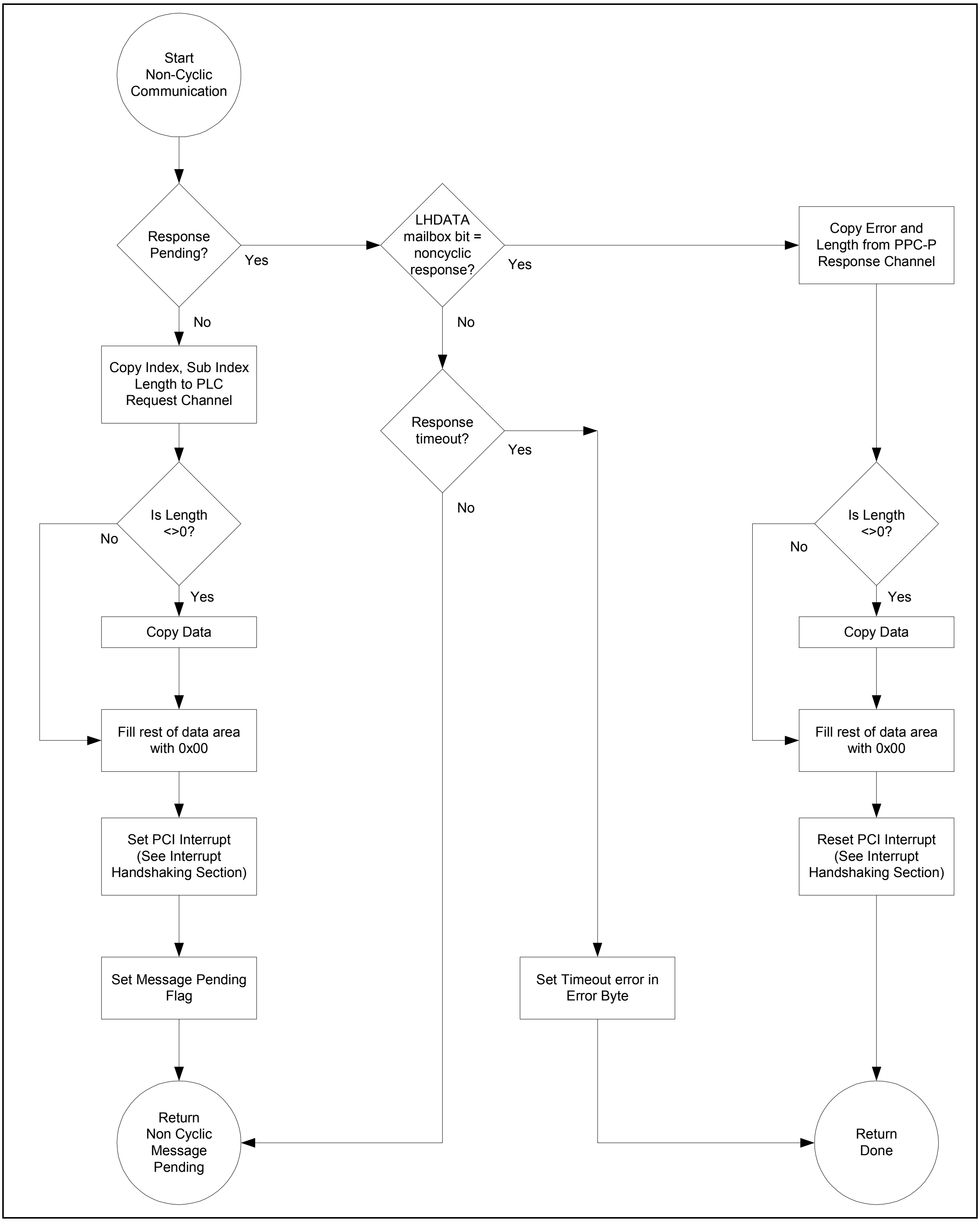


Fig. 20-15: Non-cyclic channel data transfer example

20.12 Custom Interface Examples

All examples are from an application written in C++ to communicate between Siemens WinAC RTX soft PLC V3 on VenturCom's RTX version 4.3. Some functions and defines have been provided in the VenturCom RTX API and with Siemens RTX ODK V3.0.

Note: The functions provided here are here to serve merely as examples, i.e. Bosch Rexroth does not assume any warranty for any possible compatibility problems arising in connection with future control units. Moreover, the user does not have any claim to servicing and/or extension of the published functions and structures.

Defining Structures

Before defining the functionality, the basic structures and constants should be defined.

PPC-P Card Structure

The PPC-P card structure should have members that store the PCI bus information and pointers to memory where the physical DPR is mapped. Optionally, more information can be included; in this example flags indicating various status conditions are included.

Example PPC-P card structure:

```
typedef struct _PPCP_CARD
{
 RTX_PCI pcibus;    // PPC-P PCI bus information
 PPCP_FLAGS status; // PPC-P interface status flags
 PPCP_DPR dpr;      // PPC-P DPR interface struct
} PPCP_CARD;
```

Each operating system should provide structures in its API which are used to store information about PCI cards.

PCI Bus Structure

```
// PCI Card PCI bus information
typedef struct _RTX_PCI
{
 ULONG bus;                        // Bus number of PCI card
 PCI_SLOT_NUMBER slotnumber;       // Device and Function
                                   // num of PCI card
 PCI_COMMON_CONFIG pciconfig;      // PCI configuration
                                      information
} RTX_PCI;
```

Note: PCI_SLOT_NUMBER and PCI_COMMON_CONFIG are structures included in the RTX API from VenturCom.

Status Flag Structure

The flags included show the status of 3 states operation:

- Is the lifecounter running?
- Has the initialization procedure run successfully?
- Is a noncyclic message pending? (used to ensure that only 1 request is allowed at a time)

```
typedef struct _PPCP_FLAGS
{
 BOOL ppcp_alive;   // Flag indicating the PPC-P's
                    // lifecounter is running
 BOOL ppcp_init;    // Flag indicating that the PPC-
                    // P has been initialized
 BOOL noncycpending;// Flag indicating that a non
                    // cyclic message is pending
} PPCP_FLAGS;
```

Dual Port RAM Structures

The DPR structure is defined by functionality as seen in table 9-11; PCI interrupts, cyclic communication (Register and Cyclic Channels), non-cyclic communication, Initialization status and control, and cyclic channel handshaking.

The elements in these functionality structures are pointers of the correct data type for the data accessed from the DPR. For example, the register_channel pointer in the CYCLIC_COM structure is a USHORT pointer which stores the address of the first word of the register channel.

```
// PPC-P Dual Port RAM interface
typedef struct _PPCP_DPR
{
 PCI_INTERRUPT host_to_local;    // PC -> PPC-P interrupts
 PCI_INTERRUPT local_to_host;    // PPC-P -> PC interrupts
 CYCLIC_COM plc_outputs;  // Register & Cyclic Channels
 CYCLIC_COM plc_inputs;   // Register & Cyclic Channels
 NON_CYCLIC_CO noncyclic_channel;
 PPCP_DIAGNOSTIC ppcp_diagnostics;
 PLC_INIT plc_initialization;
 PPCP_INIT ppcp_initialization;
 HANDSHAKING plc_handshaking;
 HANDSHAKING ppcp_handshaking;
} PPCP_DPR;

// Structure for PCI interrupts
typedef struct _PCI_INTERRUPT
{
 ULONG* status_control;    // PCI interrupt status and
                           // control register
 ULONG* mailbox;    // PCI interrupt (high word) and mailbox
                    // (low word)
 UCHAR* int_os;     // Operating system (windows)
                    // request of
// interrupt
 UCHAR* int_rtos;   // Real time OS (RTX) requst ofPCI
                    // interrupt
} PCI_INTERRUPT;
```

```
// Structure for cyclic data exchange
typedef struct _CYCLIC_COM
{
 USHORT* register_channel;  // Start of Register Channel with
                            // a length of
                            // LENGTH_REGISTER_CHANNEL
 ULONG* cyclic_channel;     // Start of Cyclic Channel with a
                            // length of
                            // LENGTH_CYCLIC_CHANNEL
} CYCLIC_COM;

// Structure for non-cyclic communication
typedef struct _NON_CYCLIC_COM
{
 UCHAR* plc_request;       // Start of PLC Request area of
                           // noncyclic channel with a length
                           // of LENGTH_NON_CYCLIC_CHANNEL
 UCHAR* ppcp_response;     // Start of PPC-P response area of
                           // NC channel with a length of
                           // LENGTH_NON_CYCLIC_CHANNEL
} NON_CYCLIC_COM;

// Structure for PPC-P diagnostic information
typedef struct _PPCP_DIAGNOSTIC
{
 USHORT* ppcp_error_code;  // PPCP error code number from
                           // parameter C-123
 UCHAR* diag_text;         // PPCP diag. text from parameters
                           // C-122 and C-124, w/ length of
                           // LENGTH_DIAGNOSTIC_STRING
 USHORT* Sercos_phase;     // PPCP Sercos phase from
                           // parameter C-121
} PPCP_DIAGNOSTIC;

// Structure for PLC initialization control data
typedef struct _PLC_INIT
{
 USHORT* status;    // Bit 0 = Ready, Bit 1 = Error,
                    // Bit 7 = Run
 USHORT* command;   // Initialization sequence command
                    // to PPC-P:
                    // 0x01 = startup
                    // 0x02 = program download
 USHORT* life_counter;
 UCHAR* ident_string; // Ident string to be displayed in
                      // C-0-2637 with max length of
                      // LENGTH_IDENT_STRING
 USHORT* init_phase;  // Current initialization phase
 USHORT* result;      // 1 = PPC-P Ident string is valid
} PLC_INIT;
```

```
// Structure for PPC-P initialization status data
typedef struct _PPCP_INIT
{
 USHORT* status;     // Bit 0 = Ready, Bit 1 = Error,
                     // Bit 2 = Download Not Possible,
                     // Bit 7 = Run
 USHORT* response;   // Mirrors initialization command
                     // from PLC
 USHORT* life_counter; // PPC-P life counter
 UCHAR* I dent_string; // Firmware string for GMP, with a
                       // max length of
                       // LENGTH_IDENT_STRING
 USHORT* init_phase;   // Current initialization phase
} PPCP_INIT;

// Structure for cyclic channel handshaking
typedef struct _HANDSHAKING
{
 UCHAR* output_area_request; //0x80 = request for cyclic
                             // channel area
 UCHAR* input_area_request;  //0x00 = cyclic channel is free
} HANDSHAKING;
```

Locating and Configuring PPC-P

Constants

```
// PPC-P PCI Configuration Information
#define PPCP_VENDOR_ID 0x16F2 // Bosch Rexroth Vendor ID
#define PPCP_DEVICE_ID 0x0001 // Device ID for the
                              // PPC- P011.1
#define PPCP_BASE_ADDRESS 0   // PCI DPR at Base Address [0]
#define MAX_PPCP 1            // Max number of PPC-Ps to
                              // find on PCI bus
```

Example:

```
/*##{function}###############################################
USHORT FindPpcpCard(PPCP_CARD* ppcpcard)
-----------------------------------------------------------
Description: This function scans the PCI-Bus looking for a
PPC-P card. If one is found its information is stored in a
PPCP_CARD variable.
Parameters: PPCP_CARD*
ppcpcard: A pointer to a PPC-P card structure which will be
filled with PCI-card configuration values when a PPC-P is
found.
Return: USHORT ppcpfound:        0 = No cards found.
                                 1 = successful.
###############################################{end}##*/
USHORT FindPpcpCard (PPCP_CARD* ppcpcard)
{
 ULONG devicenum;    // Logical slot number for the PCI
                     // Device
 ULONG function;     // Function number on the specified
                     // Device
 ULONG byteswritten;// Return value from
                     // RtGetBusDataByOffset
 ULONG bus = 0;      // Current bus number of the system
                     // during the search
```

```
BOOLEAN flag = TRUE; // Set to FALSE when end of system bus
                     // numbers is reached
ULONG offset = 0;    // 0 means, get all configuration data
                     // from the PCI-Device
USHORT ppcpfound = 0;// Set to 1 when PPC-P is found on PCI
                     // bus
PPCI_COMMON_CONFIG pcidata; // Pointer to a PCI
                            // configuration data structure
                            // <RTAPI.h>
PCI_SLOT_NUMBER slotnumber; // Structure for PCI slot
                            // number <RTAPI.H>
UCHAR buffer[PCI_COMMON_HDR_LENGTH];
pcidata = (PPCI_COMMON_CONFIG) buffer;
slotnumber.u.bits.Reserved = 0; // Search all the PCI
                                // busses in the PC for a
                                // PPC card.  flag will be
                                // set to FALSE, when no
                                // more PCI busses remain
                                // to be searched.
while(flag){ // Check every slot on the PCI bus for a PPC-P
    for(devicenum=0;(devicenum<PCI_MAX_DEVICES)&& flag;
                     devicenum++){
     slotnumber.u.bits.DeviceNumber = devicenum;
     // Check the function of the PCI card
for(function=0; function<PCI_MAX_FUNCTION; function++){
  slotnumber.u.bits.FunctionNumber = function;
    // Get configuration data from a PCI-Device
  byteswritten = RtGetBusDataByOffset (PCIConfiguration,
  bus,slotnumber.u.AsULONG, pcidata, offset,
  PCI_COMMON_HDR_LENGTH);
   // If no bytes were written: The PCI bus does not exist
      if(!byteswritten){
       flag = FALSE;
       break;
      }
   // No device at this slot number, skip to next slot
      if(pcidata->VendorID == PCI_INVALID_VENDORID){
        break;
      }
  /* If no interrupt pin exists for the device, it does not
     actually exist on the PCI bus, skip to next slot */
      if(pcidata->u.type0.InterruptPin == 0){
        break;
      }
  /* Check the device for correct vendor and device IDs of
     PPC-P and if the maximum number of cards has been
     reached */
 if((pcidata->VendorID == PPCP_VENDOR_ID)&&
    (pcidata->DeviceID == PPCP_DEVICE_ID)){
 // Copy PPC-P card data to structure
     ppcpcard->bus = bus;
     ppcpcard->slotnumber = slotnumber;
     ppcpcard->pciconfig = *pcidata;
 }
}
// for(function=0;funciton<PCI_MAX_FUNCTION;function++)}
// for(devicenum=0;(devicenum <PCI_MAX_DEVICES)&& flag;
// devicenum ++)
    if(flag)
     bus ++;
} // while (flag)
      return ppcpfound;
```

Mapping the DPR to Virtual Memory

Constants

```
/*     The PPC-P DPR map:

       0x0000 - 0x03FF           Reserved
       0x0400 - 0x07FF           Operations Registers
       0x0800 - 0x3FFF           Reserved
       0x4000 - 0x5FFF           Shared Memory
       0x6000 - 0x7FFF           Reserved


The PPC DPR must be mapped to the PC's virtual memory to be
used by any application.  To minimize the us of the virtual
memory, the DPR should be mapped in multiple blocks:

       Operation Registers (PCI Interrupts)
       (the following are in Shared Memory area)
       PLC Outputs  (Register and Cyclic channels)
       PLC Inputs (Register and Cyclic channels)
       Non Cyclic Channel (PLC Request/PPC Response)
       PPC Diagnostics
       Status/Control/Handshaking

*/

/* Offsets for mapping PPC-P DPR to PC's shared memory
   All areas are offset from PPCP_BASE_ADDRESS */
#define OFFSET_OPERATION_REGS 0x0400   // PCI bus interrupt
                                       // location
#define OFFSET_PLC_CYC_OUTPUTS 0x4000 // Reg/Cyclic channels
                                       // to PPC
#define OFFSET_PLC_CYC_INPUTS 0x4B00   // Reg/Cyclic channels
                                       // from PPC
#define OFFSET_NON_CYC_CHANNEL 0X5800 // PLC nc request/ PPC
                                       // response
#define OFFSET_PPCP_DIAG 0X5E70        // PPC-P diagnostics
#define OFFSET_STAT_CONT_HANDSHAKE 0X5F00
// Initialization/handshaking

/* Number of bytes to be mapped to virtual memory from DPR
   for the above areas */
#define LENGTH_OPERATION_REGS          256
#define LENGTH_PLC_CYC_OUTPUTS         1104
#define LENGTH_PLC_CYC_INPUTS          1104
#define LENGTH_NON_CYC_CHANNEL         784
#define LENGTH_PPCP_DIAG               70
#define LENGTH_STAT_CONT_HANDSHAKE     256

/* Offsets for the data areas within the virtual memory
   areas
   Offset from OFFSET_OPERATION_REGS (Bytes) */
#define OFFSET_HOST_INTERRUPT_CONT_STAT_REG   0xE4
#define OFFSET_HOST_TO_LOCAL_DATA_MAILBOX     0xE8
#define OFFSET_LOCAL_INTERRUPT_CONT_STAT_REG  0xF4
#define OFFSET_LOCAL_TO_HOST_DATA_MAILBOX     0xF8
```

```
/* Offset from both OFFSET_PLC_CYC_OUTPUTS and
   OFFSET_PLC_CYC_INPUTS  (Bytes) */
#define OFFSET_REGISTER_CHANNEL         0x00
#define OFFSET_CYCLIC_CHANNEL           0x400

// Offset from OFFSET_NON_CYC_CHANNEL (Bytes)
#define OFFSET_PLC_REQUEST              0x00
#define OFFSET_PPCP_RESPONSE            0x200

// Offset from OFFSET_PPCP_DIAG (Bytes)
#define OFFSET_PPCP_ERROR_CODE          0x04
#define OFFSET_PPCP_DIAGNOSTIC_TEXT     0x0C
#define OFFSET_Sercos_PHASE             0x48

// Offset from OFFSET_STAT_CONT_HANDSHAKE (Bytes)
#define OFFSET_PLC_STATUS               0x00
#define OFFSET_PLC_COMMAND              0x02
#define OFFSET_PLC_COUNT                0x04
#define OFFSET_MC_STATUS                0x20
#define OFFSET_MC_RESPONSE              0x22
#define OFFSET_MC_COUNT                 0x24
#define OFFSET_OUT_PLC                  0x40
#define OFFSET_OUT_PPCP                 0x41
#define OFFSET_IN_PLC                   0x42
#define OFFSET_IN_PPCP                  0x43
#define INT_REQ_OS                      0x4C
#define INT_REQ_RTOS                    0x4D
#define INT_RES_OS                      0x4E
#define INT_RES_RTOS                    0x4F
#define OFFSET_PLC_IDENT_STRING         0xA0
#define OFFSET_MC_IDENT_STRING          0xC8
#define OFFSET_PLC_INIT_PHASE           0xF0
#define OFFSET_MC_INIT_PHASE            0xF2
#define OFFSET_PLC_RESULT               0xF4

// Length of communication channels and strings
#define LENGTH_REGISTER_CHANNEL   128 // Max. Registers
                                      // transferred = 128
#define LENGTH_CYCLIC_CHANNEL     32  // Max. DWORDS
                                      // transferred = 20
#define LENGTH_NON_CYCLIC_CHANNEL 272 // Max. bytes
                                      // transferred = 272
#define LENGTH_DIAGNOSTIC_STRING  60  // Max. Chars in diag
                                      // string = 60
#define LENGTH_IDENT_STRING       40  // Max. Chars in ident
                                      // string = 40
#define MAX_LENGTH_NONCYC_DATA    255 // Max data length
                                      // allowed for SF3
```

Examples

```
/*##{function}##############################################
USHORT MapVirtualMemory( void** mappedaddress,
                         ULONG baseaddress,
                         ULONG offset,
                         ULONG busnumber,
                         ULONG length)
Description: MapVirtualMemory takes an address to the PPC-P
DPR memory area and maps a specified number of bytes to the
PC's virtual memory.
The address of the mapped memory is then stored as a void
pointer.

Parameters:  void** mappedaddress:      Pointer to the
mapped memory pointer
ULONG baseaddress:   Base address for PPC-P shared memory
ULONG offset:        Offset from base address to map
ULONG busnumber:     PCI bus number
ULONG length:        Number of bytes to map

Returns: SUCCESS = 0x00: No errors while mapping DPR area to
DPR memory

FAIL_TRANSLATE_ADDR = 0x01: Not able to translate memory
area From DPR to virtual memory

FAIL_MAP_MEMORY = 0x02: Failed to map memory to virtual
memory, and unmapping successful

FAIL_UNMAP_MEMORY = 0x03: Failed to unmap memory following a
FAIL_MAP_MEMORY;
##################################################{end}##*/

USHORT MapVirtualMemory(PVOID *mappedaddress,
                        ULONG baseaddress,
                        ULONG offset,
                        ULONG busnumber,
                        ULONG length){
 ULONG zero = 0;
 LARGE_INTEGER dpraddress;// Address of DPR on PCI bus to
                          // translate to virtual memory.
 LARGE_INTEGER translatedaddress; // Stores the translated
                                  // address of the
                                  // PPC-P DPR memory area
                                  // so that it can
                                  // be mapped to virtual
                                  // memory
 BOOL boolreturn = false;
// Format the DPR address so that it can be translated for
// mapping
 dpraddress.QuadPart = ((baseaddress) + offset);
 dpraddress.QuadPart&=~PCI_ADDRESS_MEMORY_TYPE_MASK;

// Connect the physical memory to virtual memory
 boolreturn =RtTranslateBusAddress(PCIBus, busnumber,
                                   dpraddress, &zero,
                                   & translatedaddress);
 if(!boolreturn){
  return FAIL_TRANSLATE_ADDR;
 }
```

```
// Map the DPR address space to the PC's virtual memory
 *mappedaddress=RtMapMemory(translatedaddress,length,false);

// Check if the memory was mapped correctly, if not unmap
// the memory space
 if (*mappedaddress == NULL){
   if (!RtUnmapMemory (*mappedaddress)){
     return FAIL_UNMAP_MEMORY;
   }
  return FAIL_MAP_MEMORY;
 }
 return SUCCESS;    // Memory mapped successfully
}

Function excerpt:
/*##{function}###############################################
USHORT MapPpcpMemory(PPCP_CARD* ppcpcard)

Description: MapPpcpMemory has two main purposes:
1) Map the used PPC-P Dual-Port RAM memory areas into 6
   different sections of PC virtual memory.
       a) Operations Registers
       b) PLC Outputs
       c) PPC Outputs
       d) Non-Cyclic Channel
       e) PPC-P Diagnostics
       f) Status/Control/ Handshaking
2) Address the virtual memory areas with the DPR
   descriptions

Parameters:  PPCP_CARD*   ppcpcard:    PPC-P structure for
which this function should map virtual memory.
Returns: USHORT mappingerrors: If all PPC-P DPR memory areas
map correctly, then result = 0. If a failure occurs, the
return code indicates which area failed to map.
###################################################{end}##*/

USHORT MapPpcpMemory(PPCP_CARD* ppcpcard){

void* memstart=NULL; // starting location of mapped virtual
                     // memory area.
USHORT mappingerrors=0;    // 0 = Success, bits indicate
                           // which channel failed to map
                           // correctly
USHORT maparea=0;          // If any value other than 0 is
                           // entered in this variable, a
                           // memory area did not map
                           // correctly

// *** Map Operation registers to virtual memory ***
maparea = MapVirtualMemory(&memstart,
ppcpcard->pciconfig.u.type0.BaseAddresses[0],
OFFSET_OPERATION_REGS,
ppcpcard->bus,
LENGTH_OPERATION_REGS);

 if(maparea){
   mappingerrors |= FAIL_OP_REG;
 }
 else{ // If no errors in mapping to virtual memory, then
       // assign addresses to all structure members located
      // in the Operation Register memory area Host to Local
```

```
      // PCI Interrupts: status/control register and mailbox
      // (All offsets are in bytes)

ppcpcard->dpr.host_to_local.status_control=(ULONG*)memstart
+(OFFSET_HOST_INTERRUPT_CONT_STAT_REG/4);


ppcpcard->dpr.host_to_local.mailbox=(ULONG*)(((UCHAR*)
memstart) + OFFSET_HOST_TO_LOCAL_DATA_MAILBOX);

// Local to Host PCI Interrupts: status/control register and
// mailbox (All offsets are in bytes)

ppcpcard->dpr.local_to_host.status_control=
(ULONG*)(((UCHAR*) memstart) +
OFFSET_LOCAL_INTERRUPT_CONT_STAT_REG);


ppcpcard->dpr.local_to_host.mailbox=(ULONG*)(((UCHAR*)
memstart) + OFFSET_LOCAL_TO_HOST_DATA_MAILBOX);


memstart = NULL;
maparea = 0;
}
……


// Return values for MapPpcpMemory and UnmapPpcpMemory
#define SUCCESS 0x00 // No errors while mapping PPC-P DPR
                     // to PC virtual Memory
#define FAIL_OP_REG 0x01 // Failed to (un)map Operation
                         // Regs.
#define FAIL_PLC_OUT 0x02  // Failed to (un)map PLC Output
                           // area
#define FAIL_PLC_IN 0x04   // Failed to (un)map PLC Input
                           // area
#define FAIL_NC_CH 0x08    // Failed to (un)map Non-Cyclic
                           // area
#define FAIL_PPCP_DIAG 0x10 // Failed to (un)map PPC-P Diag
                            // area
#define FAIL_STAT_CNTRL_HNDSKE 0x20 // Failed to (un)map
                                    // Status/Control/
                                    // Handshaking area
// Return values for MapVirtualMemory
// SUCCESS is used as with MapPPCPMemory. No errors while
// mapping DPR area to
// DPR memory
#define FAIL_TRANSLATE_ADDR 0x01 // not able to translate
                                 // memory area from DPR to
                                 // virtual memory
#define FAIL_MAP_MEMORY 0x02     // Failed to map memory to
                                 // virtual memory, and
                                 // unmapping successful
#define FAIL_UNMAP_MEMORY 0x04   // Failed to unmap memory
                                 // following a
                                 // FAIL_MAP_MEMORY;
```

Un-Mapping Virtual Memory

Function Example

```
/*##{function}##############################################
USHORT UnmapPpcpMemory(PPCP_CARD* ppcpcard)
-----------------------------------------------------------
Description: UnmapPpcpMemory unmaps the PPC-P card's DPR
from


Parameters:  PPCP_CARD*   ppcp_card:   PPC-P structure for
which this function should unp virtual memory.

Returns: USHORT errors:
If all PPC-P DPR memory areas unmap correctly, then result =
0. If a failure occurs, the return code indicates which area
failed to unmap.
###################################################{end}##*/

USHORT Rtx_UnmapPpcpMemory(PPCP_CARD* ppcpcard){
     USHORT errors = 0;
// Unmap Operation Register area
     if(!RtUnmapMemory ((void*)(ppcpcard-
         >pciconfig.u.type0.BaseAddresses[0] +
         OFFSET_OPERATION_REGS))){
       errors |= FAIL_OP_REG;
     }

// Unmap PLC Output area
     if(!RtUnmapMemory ((void*)(ppcpcard-
        >dpr.plc_outputs.register_channel))){
       errors |= FAIL_PLC_OUT;
     }

// Unmap PLC Input area
     if (!RtUnmapMemory ((void*)(ppcpcard-
         >dpr.plc_inputs.register_channel))){
       errors |= FAIL_PLC_IN;
     }

// Unmap Noncyclic Channel area
    if (!RtUnmapMemory ((void*)(ppcpcard-
        >dpr.noncyclic_channel.plc_request))){
       errors |= FAIL_NC_CH;
    }

// Unmap PPC-P Diagnostic area
    if (!RtUnmapMemory ((void*)(ppcpcard-
        >dpr.ppcp_diagnostics.ppcp_error_code))){
       errors |= FAIL_PPCP_DIAG;
    }

// Unmap Status/ Control/ Handshaking area
    if (!RtUnmapMemory ((void*)(ppcpcard-
        >dpr.plc_initialization.status))){
          errors |= FAIL_STAT_CNTRL_HNDSKE;
    }
     return errors;
}
```

Initializing Connection Between PLC and PPC-P

Constants

```
// Valid PPC-P firmware Typecodes
#define PPCP_GMP10 "PFM01*-GMP-10" // Release not needed -
                                   // check version level
#define FWSTR_COMPARE 13           // Number of characters
                                   // to compare in the
                                   // firmware typecode

// PCI Interrupt Status/Control Register values
#define MAILBOX_ENABLE 0x00080000  // Bit 19 enables PCI
                                   // interrupt mailbox
#define INTERRUPT_STATUS 0x00000008 // Set when interrupt
                                    // has been received
                                    // from PPC-P
#define INTERRUPT_RESET 0xFFFF0008  // Writing to status bit
                                    // clears the bit.
#define LHDATA_MASK 0x0000FFFF      // Just write mbox data,
                                    // not set new interrupt
                                    // to PLC

// PCI interrupt (| into HLdata so won't change the mailbox
// value)
#define SET_PCI_INTERRUPT 0x01000000 // Mask to set PCI
                                     // interrupt.

// PCI interrupt mailbox values (Intel format )
#define CLEAR_MAILBOX 0x00000000  // Clear Mailbox
#define NC_MSG_REQUEST 0x00000008 // Noncyclic message
                                  // request
#define INITIALIZATION 0x00000100 // Command PPC to begin
                                  // initialization process.

// PLC/PPC-P Status reg. bits
#define READY_BIT 0x0001 // Ready for further initialization
#define RUN_BIT 0x0080   // Initialization complete,
                         // Lifecounter is now monitored.
#define ERROR_BIT 0x0002 // Error during initialization
                         // sequence
#define NO_DOWNLOAD 0x0004 // PPC-P Only. Set when PPC-P is
                           // in either Sercos phase 3 or 4.
                           // New PLC program activation
                           // should only be done in Sercos
                           // phase 2.

// Initialization Commands
#define STARTUP_COMMAND 0x0002 // Command to PPC-P to begin
                               // the start up
                               // initialization process
#define SHUTDOWN_COMMAND 0x0001 // Command to PPC-P to stop
                                // monitoring the PLC's
                                // lifecounter while a new
                                // program is activated
```

```
// Return values for InitPppcpStartup SUCCESS is used as
// with MapPPCPMemory. No errors while initialization
#define CLEAR_MEMORY_FAILED 0x01 // At least one area of the
                                 // PPC-P DPR could not be
                                 // set to 0
#define PHASE_1_TIMEOUT 0x02     // 20 second timeout passed
                                 // for PPC-P to reach
                                 // initialization phase 1
#define PHASE_2_TIMEOUT 0x04     // 1 second timeout passed
                                 // for PPC-P
                                 // to reach initialization
                                 // phase 2
#define PPCP_IDENT_ERROR 0x08    // PPC-P firmware typecode
                                 // is not correct

//:*******************************************************
//: InitConnection(...)
//:-------------------------------------------------------
//: Return Value:
//:   ODK_RESULT: This user-defined value is passed to WinLC
//:-------------------------------------------------------
//: Parameters:
//: CWinLCReadData& Input: Handle for the input area of
//: WinLC, the content can be accessed with functions of the
//: CWinLCReadData class.
//: CWinLCReadWriteData& Output: Handle for the output area
//: of WinLC, can be accessed with functions of the
//: CWinLCReadWriteData class.
//:-------------------------------------------------------
//: Description:
//    User-defined function which is called from
//    the Execute method on a call to SFB65002 within WinLC.
//:*******************************************************
ODK_RESULT InitConnection
(CWinLCReadData& Input, // Handle for the input area of
                        // WinLC, the content can be
                        // accessed with functions of the
                        // CWinLCReadData class.
CWinLCReadWriteData& Output // Handle for the output area of
                            // WinLC, can be accessed with
                            // functions of the
                            // CWinLCReadWriteData class.
)
{
//: Added Subcommand Functionality Here


// The initialization of the PPC-P will take place in a
// separate thread, since it could take up to 21 seconds
// before an initialization error would be set.  In this
// function, the thread is created, started and monitored.
// When complete the result is returned to WinLC.


// Check if the PPC-P has already been initialized
if(!ppcpcard.ppcp_init){ // Check if the init thread has
                         // been created, if not create
                         // thread.
  if(initconnection == NULL){ // Create and schedule new
                              // initialization thread
    initconnection = new CInitCon();
    initconnection ->SetDelTime(CInitCon::ON_EXECUTE);
    g_Processor->ScheduleEvent(initconnection);
    return PPCP_INIT_IN_PROGRESS;
  }
```

```
// Check if the init thread is still executing. Waiting for
// it to end.

   else if(initconnection->GetStatus() ==
           CEventMsg::EXECUTING){
             return PPCP_INIT_IN_PROGRESS;
   }

// Check if the init thread has completed.

   else if(initconnection->GetStatus() ==
           CEventMsg::COMPLETE){

// Check if the initialization has failed.  Return the cause
// of the failure.

    switch(ppcpcard.initerror){
     case SUCCESS:  return ODK_SUCCESS;
     case CLEAR_MEMORY_FAILED: return PPCP_INIT_CLRMEM_FAIL;
     case PHASE_1_TIMEOUT: return PPCP_INIT_PHASE1_FAIL;
     case PHASE_2_TIMEOUT: return PPCP_INIT_PHASE2_FAIL;
     case PPCP_IDENT_ERROR: return PPCP_INIT_IDENT_FAIL;
     default: return PPCP_INTFCE_UNKNOWN_ERROR;
    } // end switch (ppcpcard.initerror)
   }//end else if (initconnection->GetStatus() ==
                   CEventMsg::COMPLETE)
   }// end if (ppcpcard.ppcp_init)

// If Card successfully initialized return success back to
// WinLC

  return ODK_SUCCESS;
}


/*##{function}##############################################
USHORT InitPpcpStartup(PPCP_CARD* ppcpcard, BOOL
checktypecode)
-----------------------------------------------------------
Description: InitPpcpStartup handles the initialization
handshaking between the PPC-P card and the PLC.  The PLC
through this function controls the process, which is
triggered on the PPC-P by an interrupt.
Following a successful initialization, the PPC-P starts its
life counter, monitors the PLC's life counter and exchanges
data.


Parameters:
PPCP_CARD* ppcpcard This PPC-P card will be initialized.
BOOL checktypecode  True activates optional check of PPC-P
typecode.
Returns:

USHORT SUCCESS: No errors, PPC-P is initialized and starts
to monitor the life counter CLR_MEM_FAIL The PPC-P memory
could not be cleared PHASE_1_FAIL No reply from the PPC-P
within the 20s timeout period for reaching  initialization
phase 1.
PHASE_2_FAIL The PPC-P failed to respond within the 1 second
time limit for reaching initialization phase 2.
###################################################{end}##*/
```

```
USHORT InitPpcpStartup(PPCP_CARD* ppcpcard,
                       BOOL checktypecode){
  ULONG timercount = 0;
  Static USHORT doonce = 0; // Used to do code only once at
                            // startup
  ULONG mailboxtemp = 0;    // Used when setting interrupts,
                            // so that mailbox values
                            // written by other processes
                            // do not get lost.
  LARGE_INTEGER sleeptime;
  LARGE_INTEGER intsleep;

  intsleep.QuadPart = HALF_MILLI_SEC; // Clear PPC-P DPR
                                      // memory (set all
                                      // areas = 0x00)
  if(!ClearPpcpDpr(ppcpcard)){
    return CLEAR_MEMORY_FAILED;
  }

// Clear the handshaking registers and mailbox if windows
// application has not done it.
  if((*ppcpcard->dpr.host_to_local.status_control &
      MAILBOX_ENABLE) == 0){
    *ppcpcard->dpr.host_to_local.int_os = NO_REQUEST;
    *ppcpcard->dpr.host_to_local.int_rtos = NO_REQUEST;
    *ppcpcard->dpr.local_to_host.int_os = NO_REQUEST;
    *ppcpcard->dpr.local_to_host.int_rtos = NO_REQUEST;

    // Clear the Local to Host Mailbox
    *ppcpcard->dpr.local_to_host.mailbox = CLEAR_MAILBOX;
   }
// Set PLC Phase 1:
    *ppcpcard->dpr.plc_initialization.init_phase = 1;
// PLC ready for further initialization
    *ppcpcard->dpr.plc_initialization.status = READY_BIT;
// Set initialization command to PPC-P to Start-up
    *ppcpcard->dpr.plc_initialization.command =
    STARTUP_COMMAND;
//Write type code for SDK to DPR
    strcpy ((char*)ppcpcard-
    >dpr.plc_initialization.ident_string, SDK_TYPECODE);

// Initialization command interrupt procedure:
// Store the command value | with the set interrupt bit and
// store in the temp variable Request interrutpt

    *ppcpcard->dpr.host_to_local.int_rtos = REQUEST;

// Check if PCI interrutpt is free, if not wait for it to
// be.
   while (*ppcpcard->dpr.host_to_local.int_os == REQUEST){
// sleep for short period of time to let windows have
// processor time to clear handshake register
RtSleepFt(&intsleep);
}
// set interrupt
  mailboxtemp = (INITIALIZATION | SET_PCI_INTERRUPT);
  // the current contents of mailbox to temp variable so
  // it is not lost
```

```
  mailboxtemp |= *ppcpcard->dpr.host_to_local.mailbox;
// write to temp mailbox back to the DPR mailbox to set
// the new interrupt

*ppcpcard->dpr.host_to_local.mailbox = mailboxtemp;
mailboxtemp = 0;

// Free interrupt
*ppcpcard->dpr.host_to_local.int_rtos = NO_REQUEST;

sleeptime.QuadPart = ONE_SECOND;
while((*ppcpcard->dpr.ppcp_initialization.init_phase != 1)){
  if(timercount == 20){
     return PHASE_1_TIMEOUT;
  }
  RtSleepFt (&sleeptime);
  timercount++;
}
  timercount = 0;

// Set PLC Phase 2
// Optional Check PPC typecode before setting init phase 2
// CHECK_TYPECODE is false by default.
if (strncmp((char*)ppcpcard-
    >dpr.ppcp_initialization.ident_string, PPCP_GMP9,
    FWSTR_COMPARE) && checktypecode){
  return PPCP_IDENT_ERROR;
}

*ppcpcard->dpr.plc_initialization.init_phase = 2;
// Wait for PPC-P to return that it is in Phase 2 (1 second
// max.)

//***** START Operation System Dependent  *****//
     sleeptime.QuadPart = HALF_SECOND; // RTX  *****//
    //***** END Operation System Dependent    *****//
     while (!(*ppcpcard->dpr.ppcp_initialization.status &=
READY_BIT))
     {
      if (timercount == 2)
      {
           return PHASE_2_TIMEOUT;
      }
      RtSleepFt (&sleeptime);
      timercount++;
      }

timercount = 0;

// Set PLC Running
*ppcpcard->dpr.plc_initialization.status |= RUN_BIT;
return SUCCESS;
}
```

```
/*##{function}################################################
BOOL ClearPpcpDpr(PPCP_CARD* ppcpcard)
---------------------------------------------------------
Description: This function clears  the DPR of a PPC-P
card.(writes zeros in every area)

Parameters:  PPPCP_CARD* ppcpcard: A pointer to a PPC card
configuration. This PPC card's DPR will be cleared.

Returns: TRUE:     Memory cleared successfully.
         FALSE:    Memory clear failed.

---------------------------------------------------------

#################################################{end}##*/

BOOL ClearPpcpDpr(PPCP_CARD* ppcpcard){

void* memsetret;

// Clear PLC Output area
memsetret = memset ((void*)ppcpcard-
                     >dpr.plc_outputs.register_channel, 0,
                    LENGTH_PLC_CYC_OUTPUTS);
if (memsetret != (void*)ppcpcard-
>dpr.plc_outputs.register_channel){
 return FALSE;
}

// Clear PLC Input area
memsetret = memset ((void*)ppcpcard-
                     >dpr.plc_inputs.register_channel, 0,
                    LENGTH_PLC_CYC_INPUTS);
if (memsetret != (void*)ppcpcard-
                  >dpr.plc_inputs.register_channel){
             return FALSE;
}

// Clear Non Cyclic Channel area
memsetret = memset ((void*)ppcpcard-
                    >dpr.noncyclic_channel.plc_request, 0,
                    LENGTH_NON_CYC_CHANNEL);
if (memsetret != (void*)ppcpcard-
                  >dpr.noncyclic_channel.plc_request){
  return FALSE;
}

// Clear PPC Diagnostics area
memsetret = memset ((void*)ppcpcard-
                     >dpr.ppcp_diagnostics.ppcp_error_code,
                     0, LENGTH_PPCP_DIAG);
if (memsetret != (void*)ppcpcard-
                  >dpr.ppcp_diagnostics.ppcp_error_code){
  return FALSE;
}
```

```
// Clear Status/Control/Handshaking area
// clear to start of Ident string and then  clear the last
// three registers directly

memsetret = memset ((void*)ppcpcard-
>dpr.plc_initialization.status,
                                              0,
OFFSET_MC_IDENT_STRING-1);
if (memsetret != (void*)ppcpcard-
    >dpr.plc_initialization.status){
 return FALSE;
}
*ppcpcard->dpr.plc_initialization.init_phase = 0x00;
*ppcpcard->dpr.plc_initialization.result = 0x00;
*ppcpcard->dpr.ppcp_initialization.init_phase = 0x00;

return TRUE;
}
```

Running and Monitoring Lifecounters

Constants

```
// Maximum number of times PPC-P life counter can miss an
update cycle
#define      MAX_MISSES_LIFECOUNTER    10
// Maximum number of cycles the PLC allows the Cyclic
channels to fail to update
#define MAX_MISSES_EXCHANGE_DATA 10     // Maximum 10 missed
cycles (two
                                        // attempts per
cycle)
```

Function

```
/*##{function}##############################################
BOOL LifeCounter(PPCP_CARD* ppcpcard)
-----------------------------------------------------------
Description: LifeCounter updates the PLC life counterchecks
that the PPC-P card life counter has been updated within a
specified number of cycles.

Parameters:  PPCP_CARD*    ppcpcard    PPC-P card

Returns:  TRUE:  PPC-P life counter running
          FALSE: PPC-P life counter has not been updated for
                 a time period defined by
                 MAX_LIFECOUNTER_MISSES

###################################################{end}##*/
```

```
BOOL LifeCounter(PPCP_CARD* ppcpcard){
static USHORTlastcyclecount = 0xFFFF;// PPC-P life counter
                                     // value from the
                                     // last time the
                                     // function was called
static USHORTmisscount = 0;  // Number of times the PPC-P
                             // life counter did not update.
                             // Test run lifecounter
if (*ppcpcard->dpr.ppcp_initialization.life_counter ==
lastcyclecount){
       if (misscount >= MAX_MISSES_LIFECOUNTER){
              return FALSE;
       }
              misscount++;
       *ppcpcard->dpr.plc_initialization.life_counter =
              *ppcpcard-
>dpr.ppcp_initialization.life_counter;

       }
else
{
       misscount = 0;
       *ppcpcard->dpr.plc_initialization.life_counter =
       *ppcpcard->dpr.ppcp_initialization.life_counter;

}
lastcyclecount = *ppcpcard-
>dpr.ppcp_initialization.life_counter;
return TRUE;
}
```

Monitoring Diagnostics

Function

```
//:*******************************************************
//: ReadPpcpDiagnostics(...)
//:-------------------------------------------------------
//: Return Value:
//:   ODK_RESULT: This user-defined value is passed to WinLC
//:-------------------------------------------------------
//: Parameters:
//:   CWinLCReadData& Input: Handle for the input area of
//:   WinLC, the content can be accessed with functions of
//:   the CWinLCReadData class CWinLCReadWriteData& Output:
//:   Handle for the output area of WinLC, can be accessed
//:   with functions of the CWinLCReadWriteData class.
//:-------------------------------------------------------
//: Description:
//    User-defined function which is called from the Execute
//    method on a call to SFB65002 within WinLC.
//
//    This function will transfer the PPC-P diagnostic data,
//    PLC command
//    data and PPC Status registers to the PLC program for
//    viewing.
//
//    It will return ODK_SUCCESS unless there is a problem
//    with one of the copy procedures.
//:*******************************************************
```

```
ODK_RESULT ReadPpcpDiagnostics
(CWinLCReadData& Input,   // Handle for the input area of
                          // WinLC, the content can be
                          // accessed with functions of the
CWinLCReadData class
CWinLCReadWriteData& Output   // Handle for the output area
                              // of WinLC, can be accessed
                              // with functions of the
                              // CWinLCReadWriteData class.
)
{
//: DONE: Added Subcommand Functionality Here
long bytecopy = 0; // temp variable used in loops for
                   // copying bytes
int count = 0;     // counter used to loop through byte
                   // arrays (text strings)

// Copy PPCP diagnostic information Error code
if(!Output.WriteS7WORD(IDerrorcode,
   *ppcpcard.dpr.ppcp_diagnostics.ppcp_error_code)){
 return PPCP_DIAG_FAIL;
}
// Diagnostic Text
for(count = 0; count < LENGTH_DIAGNOSTIC_STRING; count++){
  bytecopy = IDdiagtext+count;
  if (!Output.WriteS7BYTE(bytecopy,

  *(ppcpcard.dpr.ppcp_diagnostics.diag_text+count))){
  return PPCP_DIAG_FAIL;
 }
}
// Sercos Phase
if(!Output.WriteS7WORD(IDsercphase,

*ppcpcard.dpr.ppcp_diagnostics.Sercos_phase)){
  return PPCP_DIAG_FAIL;
}
// Copy PLC Command Registers PLC Status Register
if(!Output.WriteS7WORD(IDplcstat,

*ppcpcard.dpr.plc_initialization.status)){
  return PPCP_DIAG_FAIL;
}
// PLC Command Register
if(!Output.WriteS7WORD(IDplccmd,
   *ppcpcard.dpr.plc_initialization.command)){
   return PPCP_DIAG_FAIL;
}
// PLC life counter
// PLC Command Register
if(!Output.WriteS7WORD(IDplccount,

*ppcpcard.dpr.plc_initialization.life_counter)){
       return PPCP_DIAG_FAIL;
}
//  PLC Ident string
for (count = 0; count < LENGTH_IDENT_STRING; count++){
  bytecopy = IDplcident+count;
  if (!Output.WriteS7BYTE(bytecopy,
```

```
*(ppcpcard.dpr.plc_initialization.ident_string+count)))
                        {
                                return PPCP_DIAG_FAIL;
                        }
                }
                //  PLC Initialization phase
                if(!Output.WriteS7WORD(IDplcphase,

*ppcpcard.dpr.plc_initialization.init_phase))
                {
                        return PPCP_DIAG_FAIL;
                }
                // PLC Result
                if(!Output.WriteS7WORD(IDplcresult,

*ppcpcard.dpr.plc_initialization.result))
                {
                        return PPCP_DIAG_FAIL;
                }
                // Copy PPCP Status Registers
                // PPCP Status Regsiter
                if(!Output.WriteS7WORD(IDppcpstat,

*ppcpcard.dpr.ppcp_initialization.status))
                {
                        return PPCP_DIAG_FAIL;
                }
                // PPCP Response Register
                if(!Output.WriteS7WORD(IDppcpresp,

*ppcpcard.dpr.ppcp_initialization.response))
                {
                        return PPCP_DIAG_FAIL;
                }
                // PPCP Life counter
                if(!Output.WriteS7WORD(IDppcpcount,

*ppcpcard.dpr.ppcp_initialization.life_counter))
                {
                        return PPCP_DIAG_FAIL;
                }
                // PPCP Ident string
                for (count = 0; count < LENGTH_IDENT_STRING; count++)
                {
                        bytecopy = IDppcpident+count;
                        if (!Output.WriteS7BYTE(bytecopy,

*(ppcpcard.dpr.ppcp_initialization.ident_string + count)))
                        {
                                return PPCP_DIAG_FAIL;
                        }
                }
                // PPCP Intitailization phase
                if(!Output.WriteS7WORD(IDppcpphase,

*ppcpcard.dpr.ppcp_initialization.init_phase))
                {
```

```
            return PPCP_DIAG_FAIL;
        }

        // PPCP Result (not used)
    return ODK_SUCCESS;
}
```

Shutting Down the Connection Between PLC and PPC-P

Constants

```
// Return values for InitPppcpDownload
// SUCCESS is used as with MapPPCPMemory. No errors during
initialization
#define     DOWNLOAD_NOT_POSSIBLE     0x01   // PPC-P must
be in paramete mode

                                              // to activate new
PLC program
```

Function

```
/*##{function}################################################
##
USHORT InitPpcpDownload(PPCP_CARD* ppcpcard)
------------------------------------------------------------
Description: InitPpcpDownload stops the PPC-P from
monitoring the PLC life counter, and stops the data exchange
process.  By doing this, the PPC-P will allow a new program
to be activated on the PLC,  without issuing an life counter
error.  The new PLC program should only be allowed to be
activated with the PPC-P in parameter mode, indicated by a
bit in the PPC-P status register on the DPR.
Following a successful shutdown, the PPC-P to PLC connection
must be re-initialized.

Parameters:  PPCP_CARD*    ppcpcard

Returns: SUCCESS:   0x00   No errors, shutdown complete.
         DOWNLOAD_NOT_POSSIBLE:   0x01
PPC-P not in Parameter Mode, download not possible.

###################################################{end}##*/
USHORT InitPpcpDownload(PPCP_CARD* ppcpcard)
{
        // Check to see if the PPC-P is in Parameter mode.
If not, do not allow
        // a download (program activation)
        LARGE_INTEGERintsleep;
        intsleep.QuadPart = HALF_MILLI_SEC;

        if (*ppcpcard->dpr.ppcp_initialization.status &=
NO_DOWNLOAD)
        {
                return DOWNLOAD_NOT_POSSIBLE;
        }

        // Reset the Run bit in the PLC status bit (Stops the
PLC life counter)
        *ppcpcard->dpr.plc_initialization.status = READY_BIT;

                // Request interrutpt
```

```
        *ppcpcard->dpr.host_to_local.int_rtos = REQUEST;

        // Check if PCI interrutpt is free, if not wait for
it to be.
        while (*ppcpcard->dpr.host_to_local.int_os ==
REQUEST)
        {
                // sleep for short period of time to let
windows have
                // processor time to clear handshake register
                RtSleepFt(&intsleep);
        }
        // set interrupt
        // Set the initialization command to "shut down" to
tell PPC to stop
        // life counter monitoring and data exchange
        *ppcpcard->dpr.plc_initialization.command =
SHUTDOWN_COMMAND;
        *ppcpcard->dpr.host_to_local.mailbox =
INITIALIZATION;
        *ppcpcard->dpr.host_to_local.mailbox &=
SET_PCI_INTERRUPT;

        // Free interrupt
        *ppcpcard->dpr.host_to_local.int_rtos = NO_REQUEST;


        return SUCCESS;
}
```

Exchanging Cyclic Channel Data

Constants

```
// Handshaking Register values
#define NO_REQUEST        0x00   // Cyclic channel area not
requested
#define      REQUEST             0x80   // Request cyclic
channel area
```

Functions

```
//:*******************************************************
//: ExchangeCyclicChannel(...)
//:--------------------------------------------------------
//: Return Value:
//:   ODK_RESULT: This user-defined value is passed to WinLC
//:--------------------------------------------------------
//: Parameters:
//:   CWinLCReadData& Input: Handle for the input area of
//:   WinLC, the content can be accessed with functions of
//:   the CWinLCReadData class
//:   CWinLCReadWriteData& Output: Handle for the output
//:   area of WinLC, can be accessed with functions of the
//:   CWinLCReadWriteData class.
//:--------------------------------------------------------
//: Description:
//    User-defined function which is called from the Execute
//    method on a call to SFB65002 within WinLC. This
//    function reads in the 20 double words to exchange with
//    the PPC-P and writes them to the DPR, and reads the
//    outputs from the PPC and writes them to the PLC.  The
```

```
//    data is transferred when the DPR is free (with the h
//    handshaking registers).  If the DPR is busy, it is
//    indicated to the PLC in the status registers at the
//    end of the input data.
//
//:***
*****************************************************
ODK_RESULT ExchangeCyclicChannel
(CWinLCReadData& Input,       // Handle for the input area
                              // of WinLC, the content can
                              // be accessed with functions
                              // of the CWinLCReadData class
 CWinLCReadWriteData& Output  // Handle for the output area
                              // of WinLC, can be accessed
                              // with functions of the
                              // CWinLCReadWriteData class.)
{
//: DONE: Added Subcommand Functionality Here
// Local variables
int count = 0; // counter of elements copied to/from PLC
int bytecopy = 0;  // The counter is converted to the byte
                   // number to copy
UINT hold1 = 0;    // Data format must be correct for S7
                   // function
UINT hold2 = 0;    // Data format must be correct for S7
                   // function
WORD copyinputstatus = 0x00;  // 0x00 successful copy, 0x80
                              // copy failed
WORD copyoutputstatus = 0x00; // 0x00 successful copy, 0x80
                              // copy failed location to
                              // write copyoutputstatus word
int bytecopyinstatus = LENGTH_CYCLIC_CHANNEL* sizeof(float);
// location to write copyinputstatus word
int bytecopyoutstatus = LENGTH_CYCLIC_CHANNEL* sizeof(float)
+ sizeof(copyinputstatus);

// Request Cyclic Channel Outputs and Inputs on DPR
*ppcpcard.dpr.plc_handshaking.output_area_request = REQUEST;

// PLC Output area free?
if(*ppcpcard.dpr.ppcp_handshaking.output_area_request ==
    NO_REQUEST){
// YES - Copy data and free area, out fail word = success
// Write all 32 double words
for(count = 0; count < LENGTH_CYCLIC_CHANNEL ; count++){
// s7 functions need a byte address to start copy
bytecopy = count * sizeof(float);
// Copy data or indicate if failed
if(!Input.ReadS7DWORD(bytecopy, hold1)){
  *ppcpcard.dpr.plc_handshaking.output_area_request =
   NO_REQUEST;
   return PPCP_CYC_PLCOUTPUT_FAIL;
}
// Cast data returned to correct format
*(ppcpcard.dpr.plc_outputs.cyclic_channel + count) =
(ULONG)hold1;
}
// Mark as successful
copyoutputstatus = SUCCESS;
// Free Outputs
 }
```

```
    else
    {
       // NO - set outfail word = 0x80
       copyoutputstatus = 0x80;
    }

    // Free the Cyclic Channel Output area of DPR
    *ppcpcard.dpr.plc_handshaking.output_area_request =
NO_REQUEST;

    // Request Cyclic Channel Input area of DPR
    *ppcpcard.dpr.plc_handshaking.input_area_request =
REQUEST;

    // PLC Input area free?
    if (*ppcpcard.dpr.ppcp_handshaking.input_area_request ==
NO_REQUEST)
    {
       // YES - Copy data and free area, in fail word = 0x00
       for (count = 0; count < LENGTH_CYCLIC_CHANNEL ;
count++)
       {
       // format data for the s7 write function
              hold2 =
*(ppcpcard.dpr.plc_inputs.cyclic_channel + count);
       // s7 functions need a byte address to start copy
              bytecopy = count * sizeof(float);
       // Copy data or indicate if failed
              if (!Output.WriteS7DWORD(bytecopy, hold2))
              {
*ppcpcard.dpr.plc_handshaking.input_area_request =
NO_REQUEST;
                     return PPCP_CYC_PLCINPUT_FAIL;
              }
       }
       // Indicate successful
       copyinputstatus = SUCCESS;
    }

    else
    {
       // NO - set infail word = 0x80 (failed)
       copyinputstatus = 0x80;
    }

    // Free input area
    *ppcpcard.dpr.plc_handshaking.input_area_request =
NO_REQUEST;

    // Copy infail word
    if (!Output.WriteS7WORD(bytecopyinstatus,
copyinputstatus))
    {
       return PPCP_CYC_PLCOUTPUT_FAIL;
    }
    // Copy outfail word
```

```
    if (!Output.WriteS7WORD(bytecopyoutstatus,
copyoutputstatus))
    {
       return PPCP_CYC_PLCINPUT_FAIL;
    }

    *ppcpcard.dpr.plc_handshaking.input_area_request =
NO_REQUEST;
    *ppcpcard.dpr.plc_handshaking.output_area_request =
NO_REQUEST;

    // Return Success
    return ODK_SUCCESS;
}
```

Exchanging Register Channel Data

Constants

```
ODK_SUCCESS                      = 0x00000000,         //
General success
PPCP_CYC_PLCINPUT_FAIL= 0x80008635,      // Failed to copy
PLC inputs (Fatal)
PPCP_CYC_PLCOUTPUT_FAIL= 0x80008636,     // Failed to copy
PLC outputs (Fatal)
```

Function

```
//:***********************************************************
//: ExchangeRegChannel(...)
//:-----------------------------------------------------------
//: Return Value:
//:   ODK_RESULT: This user-defined value is passed to WinLC
//:---------------------------------------------------------
//: Parameters:
//:   CWinLCReadData& Input:       Handle for the input area
//:   of WinLC, the content can be accessed with functions
//:   of the CWinLCReadData class
//:   CWinLCReadWriteData& Output: Handle for the output
//:   area of WinLC, can be accessed with functions of the
//:   CWinLCReadWriteData class.
//:----------------------------------------------------------
//: Description:
//    User-defined function which is called from
//    the Execute method on a call to SFB65002 within WinLC.
//:***********************************************************
ODK_RESULT ExchangeRegChannel
(CWinLCReadData& Input, // Handle for the input area of
                        // WinLC, the content can be
                        // accessed with functions of the
                        // CWinLCReadData class
                        // CWinLCReadWriteData& Output
                        // Handle for the output area of
                        // WinLC, can be accessed with
                        // functions of the
                        // CWinLCReadWriteData class.
)
{
    //: DONE: Added Subcommand Functionality Here
       int count = 0;
       int bytecopy = 0;
       USHORT hold1 = 0;
       USHORT hold2 = 0;
```

```
// Read the inputs(PLC Outputs) from the PLC and send to the
// PPC-P
// Write all 20 double words
for (count = 0; count < LENGTH_REGISTER_CHANNEL ; count++){
// s7 functions need a byte address to start copy
bytecopy = count * sizeof(USHORT);
// Copy data and indicate if failed
if(!Input.ReadS7WORD(bytecopy, hold1)){
 return PPCP_CYC_PLCOUTPUT_FAIL;
}
*(ppcpcard.dpr.plc_outputs.register_channel + count) =
hold1;
}
// Read PLC Inputs from the PPC-P and send to the PLC
for(count = 0; count < LENGTH_REGISTER_CHANNEL ; count++){
hold2 = *(ppcpcard.dpr.plc_inputs.register_channel + count);
// s7 functions need a byte address to start copy
bytecopy = count * sizeof(USHORT);
// Copy data and indicate if failed
if (!Output.WriteS7WORD(bytecopy, hold2)){
 return PPCP_CYC_PLCOUTPUT_FAIL;
}
}
 return ODK_SUCCESS;
}
```

Non-Cyclic Channel Communication

Constants

```
ODK_SUCCESS = 0x00000000,         // General success
PPCP_NONCYCPENDING  = 0X80008650, // Waiting for noncyclic
                                  // response
```

Function

```
//:**********************************************************
//: NonCyclicCommunication(...)
//:----------------------------------------------------------
//: Return Value:
//:   ODK_RESULT: This user-defined value is passed to WinLC
//:----------------------------------------------------------
//: Parameters:
//:   CWinLCReadData& Input:       Handle for the input area
//:   of WinLC, the content can be accessed with functions
//:   of the CWinLCReadData class CWinLCReadWriteData&
//: Output:
//:   Handle for the output area of WinLC, can be accessed
//:   with functions of the CWinLCReadWriteData class.
//:----------------------------------------------------------
//: Description:
//    User-defined function which is called from the Execute
//:   method on a call to SFB65002 within WinLC.
// This function is used to access the non-cyclic channel to
// the PPC-P.
// It will take the request from the PLC, set the interrupt
// and then poll for the response from the PPC.  When the
// response is receive it will reset the bit in the lhdata
// mailbox and return the reply to the PLC.
//
//:**********************************************************
```

```
ODK_RESULT NonCyclicCommunication
(CWinLCReadData& Input, // Handle for the input area of
                        // WinLC, the content can be
                        // accessed with functions of the
                        // CWinLCReadData class
                        // CWinLCReadWriteData& Output
                        // Handle for the output area of
                        // WinLC, can be accessed with
                        // functions of the
                        // CWinLCReadWriteData class.)
{
//: DONE: Added Subcommand Functionality Here

USHORT datacount = 0; // Counter of data to copy time to
                      // sleep while waiting for PCI
                      // interrupt registers
LARGE_INTEGERintsleep;
intsleep.QuadPart = HALF_MILLI_SEC;
ULONG  mboxtemp = 0;  // Temp value to copy contents of pci
                      // interrupt mailbox before anding in
                      // a new request.

static UINT responsewaitcount = 0; // Counter to wait for
                                   // response

// Message Pending?
if(!ppcpcard.noncycpending){
// NO - Copy input to the PLC request area
// write index copying  each byte individually.
Input.ReadS7BYTE(REQindex,*(ppcpcard.dpr.noncyclic_channel.p
lc_request + REQindex));

Input.ReadS7BYTE(REQindex+1,*(ppcpcard.dpr.noncyclic_channel
.plc_request + REQindex + 1));
       // write subindex
       Input.ReadS7BYTE(REQsubindex,*(ppcpcard.dpr.noncyclic
_channel.plc_request + REQsubindex));

       Input.ReadS7BYTE(REQsubindex+1,*(ppcpcard.dpr.noncycl
ic_channel.plc_request +
REQsubindex + 1));

       // write length
       Input.ReadS7BYTE(REQlength,

*(ppcpcard.dpr.noncyclic_channel.plc_request + REQlength));
       // write data
for (datacount = 0; datacount <
     *(ppcpcard.dpr.noncyclic_channel.plc_request +
     REQlength); datacount++)
       {
           Input.ReadS7BYTE(REQdata + datacount,
              *(ppcpcard.dpr.noncyclic_channel.plc_request +
REQdata + datacount));
       }
// Fill the rest of the channel with 0x00

while (datacount < MAX_LENGTH_NONCYC_DATA)
{
*(ppcpcard.dpr.noncyclic_channel.plc_request + REQdata +
datacount) = 0x00;datacount++;
}
// set interrupt to PLC
```

```
// Set interrupt request
*ppcpcard.dpr.host_to_local.int_rtos = REQUEST;

// Check if PCI interrutpt is free, if not wait for it to
// be.
while (*ppcpcard.dpr.host_to_local.int_os == REQUEST)
{
// sleep for short period of time to let windows have
// processor time to clear handshake register
   RtSleepFt(&intsleep);
}
// set interrupt
// Set the initialization command to "shut down" to tell PPC
// to stop life counter monitoring and data exchange

mboxtemp = *ppcpcard.dpr.host_to_local.mailbox;
mboxtemp |= NC_MSG_REQUEST |SET_PCI_INTERRUPT;
*ppcpcard.dpr.host_to_local.mailbox = mboxtemp;
// Free interrupt
*ppcpcard.dpr.host_to_local.int_rtos = NO_REQUEST;
mboxtemp = 0;

// set pending flag and return message pending response to
// PLC

ppcpcard.noncycpending = true;
// reset response counter to 0
responsewaitcount = 0;
return PPCP_NONCYCPENDING;
}
else
{
// YES - Check for response from PPC
if (!(*ppcpcard.dpr.local_to_host.mailbox & NC_MSG_REQUEST))
// NO - Return message pending to PLC
{
 // check if timeout condition has been hit
    if (responsewaitcount >= NONCYC_MAXCYCLES)
    {
       // set error
       Output.WriteS7BYTE(RESerror,NCYC_TIMEOUT_ERROR);
       // reset pending flag
       ppcpcard.noncycpending = false;
       // return
       return ODK_SUCCESS;
    }
    responsewaitcount++;
    return PPCP_NONCYCPENDING;
}
else
{
    // YES - Write response to PLC
    // write error bit
   Output.WriteS7BYTE(RESerror,
*(ppcpcard.dpr.noncyclic_channel.ppcp_response + RESerror));
    // write length
   Output.WriteS7BYTE(RESlength,
```

```
   *(ppcpcard.dpr.noncyclic_channel.ppcp_response +
RESlength));

// write data
for(datacount = 0; datacount <
*(ppcpcard.dpr.noncyclic_channel.ppcp_response + RESlength);
datacount++)
{
Output.WriteS7BYTE(RESdata + datacount,
*(ppcpcard.dpr.noncyclic_channel.ppcp_response + RESdata +
datacount));
 }
    // Reset the rest of the bytes to 0
    while (datacount <MAX_LENGTH_NONCYC_DATA)
    {
      Output.WriteS7BYTE(RESdata + datacount,0x00);
      datacount++;
    }

// Reset bit in lhdata mailbox and message pending flag
    ppcpcard.noncycpending = false;
// Request lhdata mailbox
 *ppcpcard.dpr.local_to_host.int_rtos = REQUEST;
  // Wait until Windows ap is done.
while (*ppcpcard.dpr.local_to_host.int_os == REQUEST)
{
// sleep for short period of time to let windows have
// processor time to clear handshake register
RtSleepFt(&intsleep);
}

// Read lhdata into temp variable
   mboxtemp = *ppcpcard.dpr.local_to_host.mailbox;
// Reset interrupt
// clear interrupt bit only
   mboxtemp ^= NC_MSG_REQUEST;
// write temp mailbox value back to PPC-P mailbox, masked
// so won't set new interrupt to PLC
   *ppcpcard.dpr.local_to_host.mailbox = mboxtemp &
// LHDATA_MASK;

// clear lhdata mailbox request
   *ppcpcard.dpr.local_to_host.int_rtos = NO_REQUEST;
    mboxtemp = 0;
 }
}
 //Return success
  return ODK_SUCCESS;
}
```


Rexroth Bosch Group

21 Rexroth BTC 06 Interface

21.1 Overview

Rexroth BTC 06 **H**uman **M**achine **I**nterface unit is used to interface with the control, providing the operator with a variety of functionality. The operator can view and modify parameters, jog axes, and interface with machine operations using terminal emulation software. Using ScreenManager software, a machine builder can create customized screens that are specific to an application.

For information regarding the setup and use of ScreenManager, refer to the following manuals:

- ScreenManager V01 Functional Description
  - DOK-SUPPL*-SCM*PROG***-FK02-EN-P
- ScreenManager V01 Application Manual
  - DOK-SUPPL*-SCM*BEDIEN*-AW02-EN-P

Note: This chapter will focus on the BTC 06 using VT100 Terminal emulation software (SWA-BTC06*-VT-01VRS-MS-C1.44). Refer to the *Rexroth VisualMotion 11 Project Planning* manual for details regarding hardware setup.

21.2 BTC 06 Teach Pendant Screens

The BTC 06 is a hand-held instrument with 16 x 40 character display and a 48-key sealed membrane keypad. The pendant provides a convenient operation and position programming interface for Rexroth VisualMotion control. The BTC 06 Teach Pendant gives users a hand held operating interface which allows them to:

- Select operating modes and axis jogging
- Access multi-level menus for functions
- Teach and edit motion control points, events and variables; edit parameters
- Select and activate user programs

Each category of functions has its own set of menus. The following function categories are available through the pendant BTV06 Main Menu:

```
         GPP MAIN MENU
               A
        PSM01*-GPP-11Vxx-M

     F1 Program Menu
     F2 Table Edit Menu
     F3 Jog Menu
     F4 Control Menu
     F5 Register I/O Menu
     F6 Parameter Menu
     F7 Security Menu
     F8 Diagnostic Menu
```

Menu Map

The following chart maps the submenus and menu links that are found within the main menu. Some menus have direct links to diagnostics, parameters and I/O registers.

Note: Pressing the ESC key will backtrack the Teach Pendant and display the previously viewed screen until it reaches the main menu.

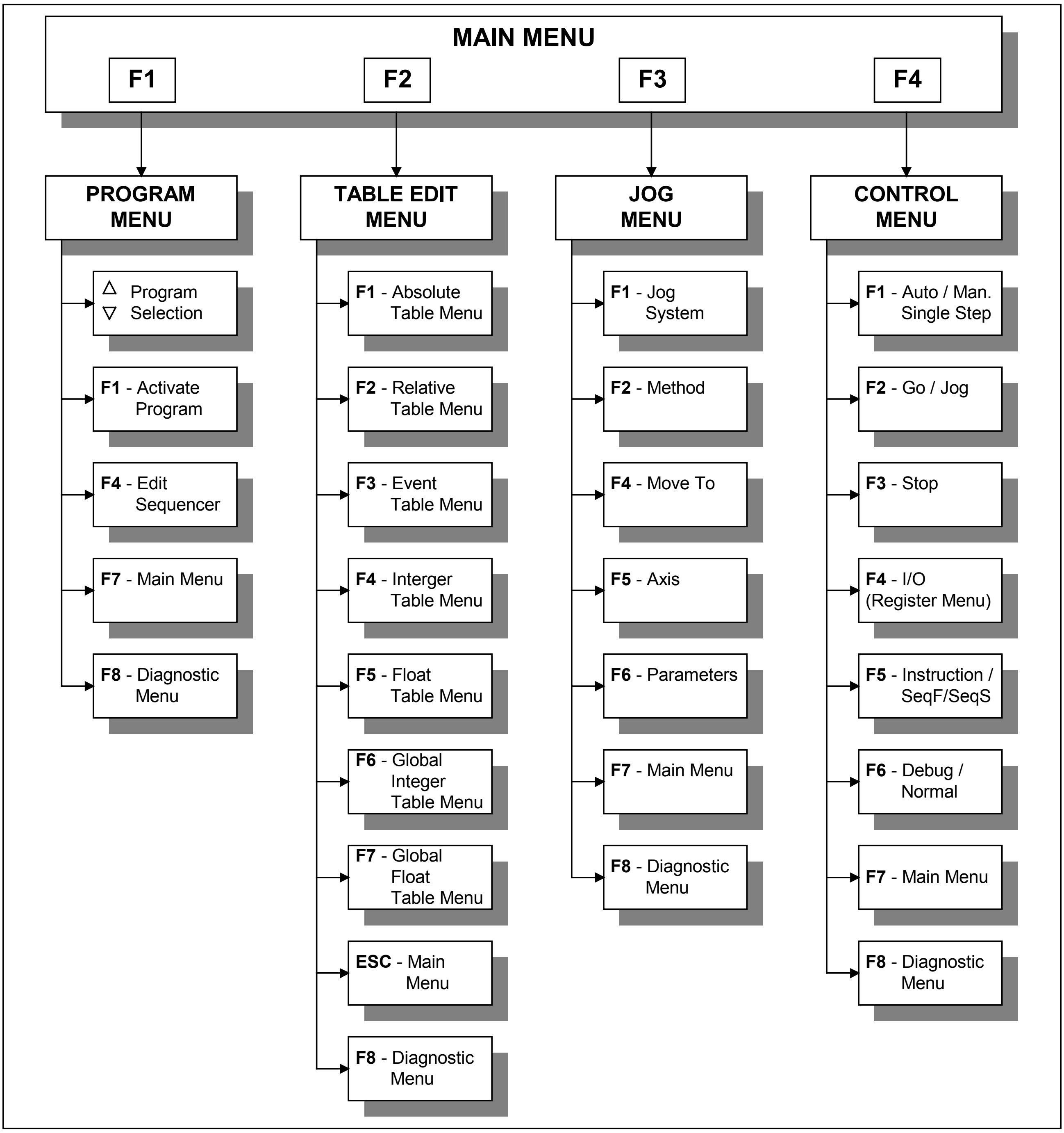


Figure 21-1: Menu Map (F1-F4)

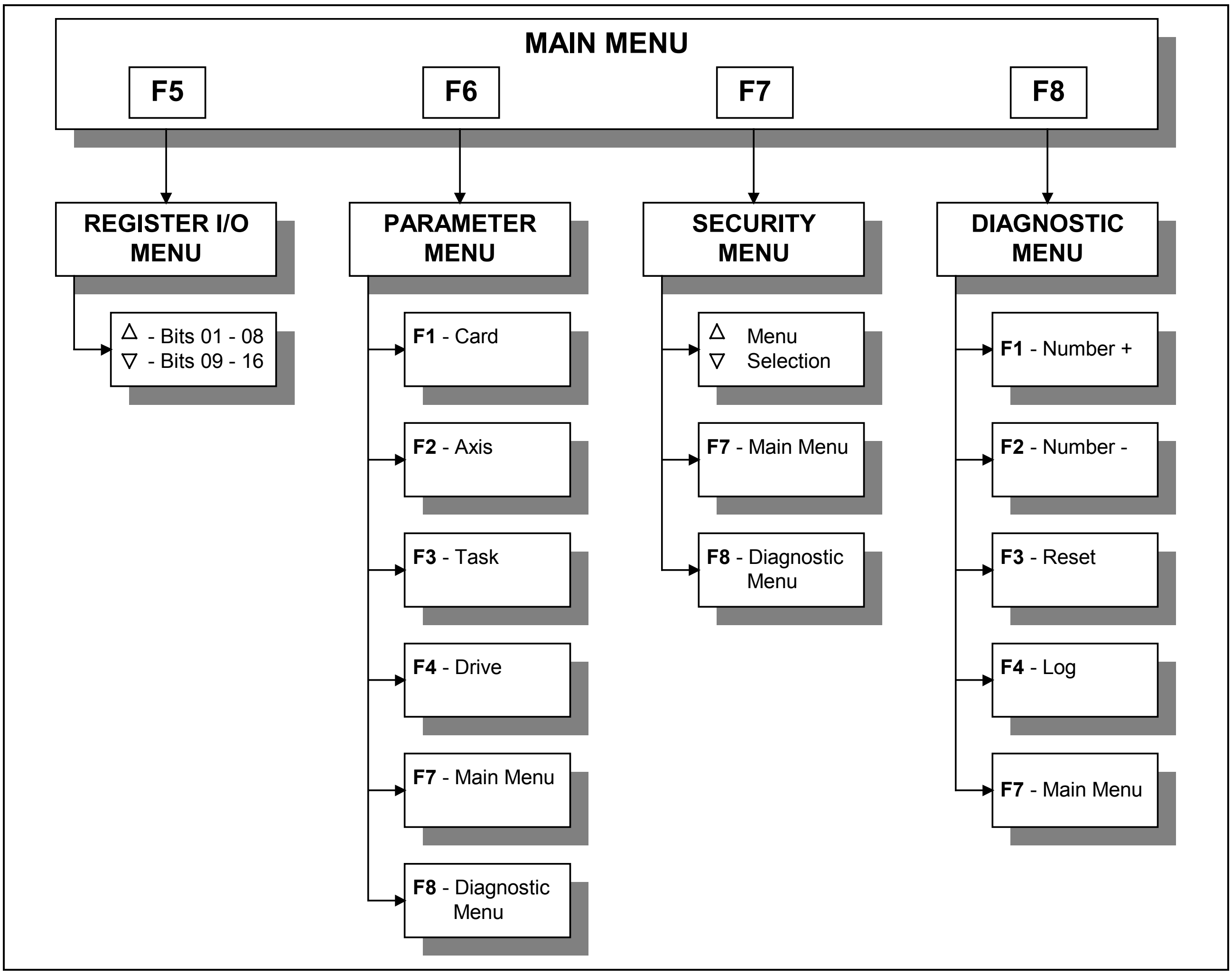


Figure 21-2: Menu Map (F5-F8)

21.3 BTC 06 Teach Pendant Setup

When the Teach Pendant is enabled, the following registers and bits are forced at all times by the BTC 06. VisualMotion Toolkit provides a register forcing capability that allows a Host system to directly change the state of individual I/O register bits overriding both the physical I/O and the I/O mapping.

Note: To use the pendant function keys, as well as the pendant edit features, the System Control Register 1, bit 14 (Pendant Enable) must be set to 1

Task A-D, Control Register 2-5:	bit 1	Mode Auto nManual
	bit 4	Single Step
	bit 6	Cycle Start/Resume
	bit 7	nTask Stop
	bit 12	Step Sequence Step
	bit 13	Step Sequence Function

Registers 98 and 99 define blocking bits for task A, B, C and D. The bits in the register can disable Teach Pendant control of the selected function for the corresponding tasks A-B, C-D. The following functions can be blocked:

Reg. - Bit	Description	Reg. - Bit	Description
98-1	Block Task A Manual	99-1	Block Task C Manual
98-2	Block Task A Auto	99-2	Block Task C Auto
98-3	Block Task A Step	99-3	Block Task C Step
98-4	Block Task A Jog	99-4	Block Task C Jog
98-5	Block Task A Entry	99-5	Block Task C Entry
98-6	Block Task A Teach	99-6	Block Task C Teach
98-9	Block Task B Manual	99-9	Block Task D Manual
98-10	Block Task B Auto	99-10	Block Task D Auto
98-11	Block Task B Step	99-11	Block Task D Step
98-12	Block Task B Jog	99-12	Block Task D Jog
98-13	Block Task B Entry	99-13	Block Task D Entry
98-14	Block Task B Teach	99-14	Block Task D Teach

If a block bit is set, its corresponding function is blocked. If a user selects the function, an error message is issued by the BTC 06.

The BTC 06 parameters are automatically preset to the following specifications:

Menu	Item	Default Setting
Serial Communication	Baud Rate	9600 (Fixed)
	Parity	None
	Data and Stop Bits	8, 1
	Display Serial Errors	Yes
	Audible Serial Errors	Yes
	Support for XON/OFF	Yes
Display	Display CTL Characters	No
	Display ESC Characters	No
	Cursor Visible	Yes
	Auto Line Wrap	No
	New Line on CR	Yes
	Display Self-Test	No
	Backlit Level	7
	Backlit On	Yes
Keyboard	Local Echo	No
	Key Repeat	Off
	Audible Keys	No
	Simplified KB	Yes

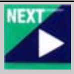
Note: When the Teach Pendant is initializing, it automatically sets the baud rate to 9600.

21.4 BTC 06 Keyboard Operation

The following defines the keys for the Teach Pendant:

Key	Action
F1	Soft key defined by active menu
F2	Soft key defined by active menu
F3	Soft key defined by active menu
F4	Soft key defined by active menu
F5	Soft key defined by active menu
F6	Soft key defined by active menu
F7	Soft key defined by active menu
F8	Soft key defined by active menu
L1 R1	left and right refresh of the BTC 06 screen

Key	Action
HELP	First press function key help, then press item help (only available for Parameter Menu)
MAIN MENU	Display all main menu functions
A+ A-	Jog A coordinate plus/minus
B+ B-	Jog B coordinate plus/minus Main Menu: respectively turns backlighting on and off
C+ C-	Jog C coordinate plus/minus
X+ X-	Jog X coordinate plus/minus
Y+ Y-	Jog Y coordinate plus/minus
Z+ Z-	Jog Z coordinate plus/minus
0	numeric key
STU 1	numeric key and letter combination (use the shift key to access letters)
VW 2	numeric key and letter combination
XYZ 3	numeric key and letter combination
JKL 4	numeric key and letter combination
MNO 5	numeric key and letter combination
PQR 6	numeric key and letter combination
ABC 7	numeric key and letter combination
DEF 8	numeric key and letter combination
GHI 9	numeric key and letter combination
.	decimal point
TEACH	Teach current position to absolute point table
TASK	Select a user task
EDIT	Clear field of current item and allow editing
↑ ↓	page up / page down
▲ ▼	up and down arrows
DEL ◀	delete and left arrow (use the shift key to access delete)

Key	Action
	next and right arrow (use the shift key to access next)
	plus and minus (use the shift key to access plus)
	Terminate current operation or return to previous menu
	Confirm entry
	Shift key

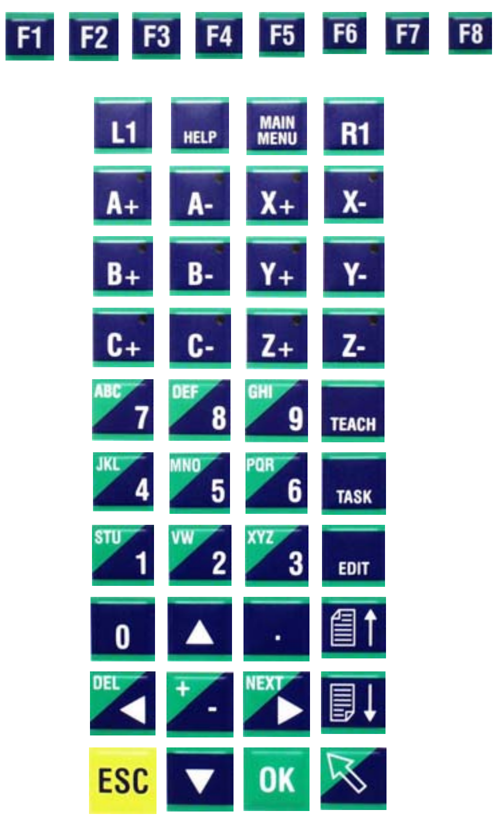
Keyboard Map

The BTC 06 keyboard is mapped to register 95, 96 and 97. The figure below and to the right outlines the register and bit location in the following format:

Register - Bit

Example: **95 - 01** ; *key is mapped to register 95, bit 01*

When a key is pressed its corresponding bit turns on and remains on for as long as the key is pressed.



95-01	95-02	95-03	95-04	95-05	95-06	95-07	95-08

97-13	95-09	97-14	97-15
96-05	96-06	96-07	96-08
96-13	96-14	96-15	96-16
97-05	97-06	97-07	97-08
95-10	95-11	95-12	95-13
96-02	96-03	96-04	95-14
96-10	96-11	96-12	96-09
97-03	95-11	97-02	95-15
96-02	97-04	96-04	95-16
96-01	96-11	97-01	

This shift key is not mapped to a register.

Cursor Control and Editing

The cursor may be moved up or down, left or right by pressing the corresponding arrow key. The left and right arrow keys double as delete and next, respectively. To edit an item, position the cursor over it and press the EDIT key. Doing so clears the field used by the item allowing a new value to be entered. Pressing OK terminates the editor and enters the new value into the system. Sometimes the cursor can be positioned on an item but the EDIT key does nothing when pressed. In this case the item cannot be edited. The cursor may be positioned there for another reason, such as item selection or viewing.

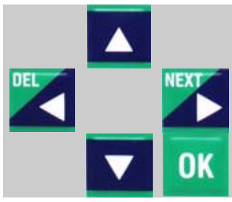


Fig. 21-3: Cursor Control and Editing

Number or Letter Selection

The number keys labeled 1 - 9 also double as letter keys when the shift  key is pressed. To select the second or third letter contained in the upper left position of the key, the shift key must be pressed and held. If the shift key is pressed and not held, only the first letter will be selected and the key will then default to the number specified.

Fig. 21-4: Using the Shift key

Jogging Control

Press the *coordinated jog keys* (X+, X-, Y+, Y-, Z+, Z-) to jog in World, Joint or Tool coordinates. For robotics, the (A+, A-, B+, B-, C+, C-) keys function as Row, Pitch and Yaw in coordinated motion. The jog keys are active only while in the Jog Menu (***F3***). If other coordinated axes are defined in other tasks, then that task must be activated in order to jog from the Teach Pendant. When in single axis mode within the Jog Menu, the (A+, A-) keys light up and are used to jog the axis.

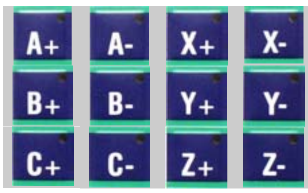


Fig. 21-5: Jogging Control

Task Control

Press the TASK  key to display the task menu. Use the arrow keys to position the cursor in the desired task and press OK, then press ESC to return to the previous menu.

Teach Control

The Teach  key allows the user to store the current position (during a coordinated jog) into the Absolute Point Table. The table point number will flash indicating that point has been recorded in the table.

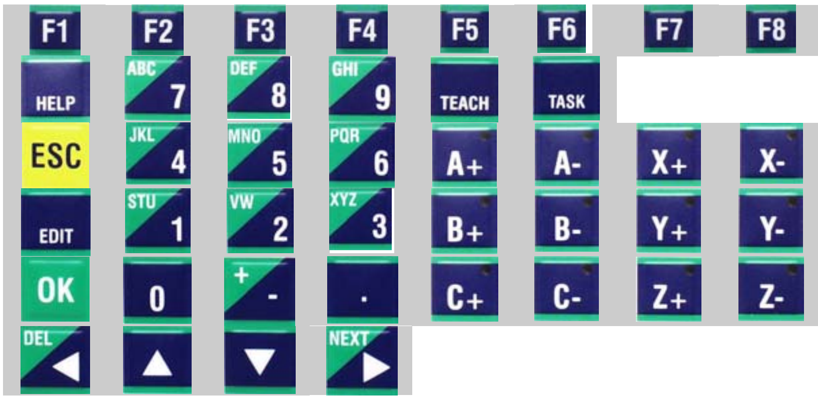


Fig. 21-6: Teach Control

21.5 F1 Program Menu

The Program menu allows pendant selection and activation of any of the programs that have been downloaded to the control.

Each program consists of one to four user tasks (A, B, C, and D), and the associated Absolute and Relative Point Tables, Event Table, and Variable Tables. Activating a new program replaces the current four motion tasks and tables with the tasks and tables for the new program selection.

The Program menu consists of downloaded user programs with the following information:

- Program number (1-10)
- Program name
- Program date
- Time
- Program size

```
             GPP PROGRAM MENU

01 SEQ          08/24/00  15:39:27    1572
02 SEQ1         08/25/00  10:20:15    3452
03 AB1          08/29/00  16:20:00    1152
04 AB2          08/29/00  16:20:00    3344
05 Sequencer    08/30/00  07:15:00    2888
00
00
00
00
00

F1    F2    F3    F4    F5    F6    F7    F8
Actv              Edit              Main  Diag
```

The up and down arrow keys move the cursor to select a program. Pressing **F1** activates the selected program.

Note: The currently active program must not be running when activating another program.

Sequencer Editing (F4)

The **F4** key (Edit) only applies to programs which contain Sequencers. Pressing **F4** allows the user to edit the Sequencer list, steps and functions of the selected program.

Sequence List Menu

The first screen that appears after pressing **F4** in the *GPP Program Menu* is the *Sequence List Menu*. Use the arrow keys to navigate with the cursor to select the desired *Sequence List*. Press **F4** again to edit the contents of the selected list name within the *Sequence Edit Menu*.

```
             SEQUENCE LIST MENU
Program: Sequencer
01 PICk_AND_PLACE
00
00
00
00
00
00
00
F1   F2         F4              F7   F8
PgUp PgDn       Edit            Main Diag
```

The name of each list can also be edited. Position the cursor at the end of the list name and press the **Edit** key. The letters of the alphabet are located within the numbered keys. These letters can only be accessed when used in conjunction with the **Shift** key. Use the **F1** key to delete characters to the left of the cursor. Use the **Shift** key to Select **Shift On** and **Off**. This allows you to toggle the keyboard map between numbers and letters. Refer to Number or Letter Selection on page 21-8 for details.

This editing process is functional within all of the following Sequencer menus.

The Sequence Edit Menu

The *Sequence Edit Menu* displays all of the steps within the selected Sequence. Use the arrow keys to navigate the cursor to the desired Sequence Step. Press **F4** again to edit the contents of that Step within the *Step Table Edit Menu*. Press **F3** to cut the selected Sequence Step. Press **F6** to paste a Sequence Step in the current cursor position

```
             SEQUENCE EDIT MENU
Program:  Sequencer
Sequence: PICK_AND_PLACE
01 Home_All_Axes
02 Set_Max_Values
03 Pick_Position
00
00
00
00
F1   F2   F3   F4   F5   F6   F7   F8
PgUp PgDn Cut  Edit Ins  Past Main Diag
```

Table List Menu

Pressing **F5** (Ins) from the *Sequence Edit Menu* will open the *Step List Menu*. This menu contains a list of all the step tables available within the selected Sequence. Use the arrow keys to navigate the cursor to the desired *Step Table*. Press OK to insert that function into the previous *Sequence*.

```
              STEP LIST

     Press OK to insert

01 Pick_Position
02 Set_Max_Value
03 Home_All_Axes
00
00
00
00
00

F1   F2
PgUp PgDn
```

The Step Table Edit Menu

The *Step Table Edit Menu* displays all the functions within the selected Sequence Step. Use the arrow keys to navigate the cursor to the desired Sequence *Function*. Press **F4** again to edit the contents of that function within the *Function Edit Menu*. Press **F3** to cut the selected function. Press **F6** to paste a function in the current cursor position

```
          STEP TABLE EDIT MENU

Program:      Sequencer
Sequence:     Initialize Sequencer
Step:         Home_All_Axes
Empty Slots:  00011

01 Disable_Clamp_Motion
02 Chk_Mold_Open
03 Permit_Eject_Back
04 Chk_Ejectors_Back
05 Home_Axes

F1   F2   F3   F4   F5   F6   F7   F8
PgUp PgDn Cut  Edit Ins  Past Main Diag
```

Function List

Pressing **F5** (Ins) from the *Step Table Edit Menu* will open the *Function List Menu*. This menu contains a list of all the functions available within the selected Sequence. Use the arrow keys to navigate the cursor to the desired *Function*. Press OK to insert that function into the previous *Step Table*.

```
              FUNCTION LIST

     Press OK to insert

01 HOME_AXIS
02 INIT_POS_VELOCITY
03 CHK_EJECTORS_RETRACT
04 CLEAR_SYSTEM_TIMERS
05 DWELL
06 PERMIT_EJECT_BACK
00
00

F1   F2
PgUp PgDn
```

Function Edit Menu

The *Function Edit Menu* contains a list of all the arguments and their corresponding values. Use the arrow keys to navigate with the cursor to the desired *Function*. Press **F4** again to edit the values assigned to the arguments of that function.

```
           FUNCTION EDIT MENU

Program:  Sequencer
Sequence: Initialize Sequencer
Step:     Home_All_Axes
Function: Home_Axes

01 AXIS_NUMBER                         2
02 HOME_OFFSET_POSITION           0.0000
03 SET_HOME_POSITION              0.0000
00
00

F1             F4             F7   F8
Save           List-Edit      Main Diag
```

21.6 F2 Table Edit Menu

The Table Edit menu allows editing of the Absolute and Relative Point Tables, the Event Table, and the Integer and Float variable Tables.

```
         GPP TABLE EDIT MENU

     F1 Absolute Table Menu
     F2 Relative Table Menu
     F3 Event Table Menu
     F4 Integer Table Menu
     F5 Float Table Menu
     F6 Global Integer Table Menu
     F7 Global Float Table Menu


                              ESC   F8
                              Main  Diag
```

Absolute Table Menu (F1)

The Absolute Point Table Edit menus permit editing taught or programmed points.

```
           ABSOLUTE TABLE

NUM NAME
001 Part_Pickup
002 Regrip
003 Leave_Part
004 ABS[4]
005 ABS[5]
006 ABS[6]
007 ABS[7]
008 ABS[8]
009 ABS[9]
010 ABS[10]

F1   F2   F3   F4   F5   F6  F7   F8
PgUp PgDn Home End  Edit Jog Main Diag
```

Select a point by moving the cursor up and down with the arrow keys. Pressing F5 (Edit) will bring up the following menu:

```
               ABSOLUTE POINT EDIT

ABS[0001]   ABS[1]

       2.000 X               3.000 Roll
       2.000 Y               1.000 Pitch
       3.000 Z               0.500 Yaw
       0.000 Blend              01 Elbow

          10 Speed               0 Event 1
           5 Accel               0 Event 2
           5 Decel               0 Event 3
           2 Jerk                0 Event 4

F1   F2   F3   F4         F6   F7   F8
Inpt DcPt Home End        Jog  Main Diag
```

X	X coordinate of the point
Y	Y coordinate of the point
Z	Z coordinate of the point
Blend	Blend Radius
Roll	Roll angle
Pitch	Pitch angle
Yaw	Yaw angle
Elbow	Elbow state
Speed	Speed Percentage (of task maximum)
Accel	Acceleration Percentage (of task maximum)
Decel	Deceleration Percentage (of task maximum)
Jerk	Jerk Limiting Percentage (0 trapezoid, 100 s-shape, 50 between)
Event 1	First event for the point
Event 2	Second event for the point
Event 3	Third event for the point
Event 4	Fourth event for the point (This value represents an event number from the event table) Refer to **Event Table** on page 21-14

Relative Table Menu (F2)

The Relative Point Table Edit menus permit editing of points either taught or programmed.

```
               RELATIVE TABLE

NUM NAME
001 REL[1]
002 REL[2]
003 REL[3]
004 REL[4]
005 REL[5]
006 REL[6]
007 REL[7]
008 REL[8]
009 REL[9]
010 REL[10]

F1   F2   F3   F4   F5   F6   F7   F8
PgUp PgDn Home End  Edit Jog  Main Diag
```

Select a point by moving the cursor up and down with the arrow keys. Pressing F5 (Edit) will bring up the following menu:

```
              RELATIVE POINT EDIT
REL[0001]  REL[1]

       2.000 Drive_1       3.000 Drive_4
       2.000 Drive_2       1.000 Drive_5
       3.000 Drive_3       0.500 Drive_6
       0.000 Blend            01 Elbow

          10 Speed             0 Event 1
           5 Accel             0 Event 2
           5 Decel             0 Event 3
           2 Jerk              0 Event 4

F1   F2   F3   F4        F6   F7   F8
Inpt DcPt Home End       Jog  Main Diag
```

X	X coordinate of the point
Y	Y coordinate of the point
Z	Z coordinate of the point
Blend	Blend Radius
Roll	Roll angle
Pitch	Pitch angle
Yaw	Yaw angle
Elbow	Elbow state
Speed	Speed Percentage (of task maximum)
Accel	Acceleration Percentage (of task maximum)
Decel	Deceleration Percentage (of task maximum)
Jerk	Jerk Limiting Percentage (0 trapezoid, 100 s-shape, 50 between)
Event 1	First event for the point
Event 2	Second event for the point
Event 3	Third event for the point
Event 4	Fourth event for the point (This value represents an event number from the event table) Refer to **Event Table** on page 21-14

Event Table Menu (F3)

The Event Table Edit menu allows pendant editing of the events associated with each task in the Event Table.

The currently selected task determines the portion of the event table allowed to be viewed through the Teach Pendant.

```
               EVENT TABLE

NUM ST TY RF      ARG FUNCTION
001 01 06 00     20.0 Pressure_Switch
002 01 06 00     40.0 Change_Speed
003 01 09 00     60.0 Evt_Fn_1
004 00 00 00      0.0 NONE
005 00 00 00      0.0 NONE
006 00 00 00      0.0 NONE
007 00 00 00      0.0 NONE
008 00 00 00      0.0 NONE
009 00 00 00      0.0 NONE
010 00 00 00      0.0 NONE

F1   F2   F3         F5          F7   F8
PgUp PgDn Home       Repl        Main Diag
```

St **The Event's status:**
0 = inactive
1 = pending
2 = queued
3 = executing
4 = done

Ty **Event type**:
0 = event inactive
1 = repeating timer
2 = time in coordinated path
3 = percent in coordinated path
4 = single axis distance
5 = repeating axis position (rotary)
6 = task input transition
9 = feedback capture
10=I/O register event
11=PPC-R X1 input events

Rf **Event Reference**:
0 = start of segment
1 = end of segment

Arg **Argument for the event**
(milliseconds if time based, or percent of path and axis distance)

Function Task ID and Event number

Integer Table Menu (F4)

This menu allows for viewing and editing integers. Variables can be changed by any task at any time. It is possible, therefore, for editing to be in conflict with a motion task. In this instance, unexpected results may occur. It is at the discretion of the operator to determine the usefulness of such an operation.

```
                    INTEGER TABLE

00001 Pointer_1                             20.0
00002 Pointer_2                             40.0
00003 Timer_1                               60.0
00004 Timer_2                               80.0
00005 Operation Type                        00.0
00006 I[6]                                     0
00007 I[7]                                     0
00008 I[8]                                     0
00009 I[9]                                     0
00010 I[10]                                    0


F1   F2   F3                       F7   F8
PgUp PgDn Fmt                      Main Diag
```

Display Format

Pressing **F3** toggles the display between decimal (20.0) and hexadecimal (0x00000014) notation.

Floating Table Menu (F5)

This menu allows for viewing and editing of float variables. Variables can be changed by any task at any time. Therefore, its possible, for editing to be in conflict with a motion task. In this instance, unexpected results may occur. It is at the discretion of the operator to determine the usefulness of such an operation.

```
                FLOATING TABLE

00001 F[1]                          0.0000
00002 F[2]                          0.0000
00003 F[3]                          0.0000
00004 F[4]                          0.0000
00005 F[5]                          0.0000
00006 F[6]                          0.0000
00007 F[7]                          0.0000
00008 F[8]                          0.0000
00009 F[9]                          0.0000
00010 F[10]                         0.0000


F1   F2   F3                   F7   F8
PgUp PgDn Fmt                  Main Diag
```

Display Format

Pressing **F3** toggles the display between floating fixed (100.000) and scientific (1.000e+02) notation.

Global Integer Table Menu (F6)

This menu allows for viewing and editing of global integer variables. Variables can be changed by any task at any time. It is possible, therefore, for editing to be in conflict with a motion task. In this instance, unexpected results may occur. It is at the discretion of the operator to determine the usefulness of such an operation.

```
            GLOBAL INTEGER TABLE

00001 GI[1]                              0
00002 GI[2]                              0
00003 GI[3]                              0
00004 GI[4]                              0
00005 GI[5]                              0
00006 GI[6]                              0
00007 GI[7]                              0
00008 GI[8]                              0
00009 GI[9]                              0
00010 GI[10]                             0


F1   F2   F3                   F7   F8
PgUp PgDn Fmt                  Main Diag
```

Display Format

Pressing F3 toggles the display between decimal (20.0) and hexadecimal (0x00000014) notation.

Global Floating Table Menu (F7)

This menu allows for viewing and editing of global float variables. Variables can be changed by any task at any time. It is possible, therefore, for editing to be in conflict with a motion task. In this instance, unexpected results may occur. It is at the discretion of the operator to determine the usefulness of such an operation.

```
           GLOBAL FLOATING TABLE

00001 GF[1]                         0.0000
00002 GF[2]                         0.0000
00003 GF[3]                         0.0000
00004 GF[4]                         0.0000
00005 GF[5]                         0.0000
00006 GF[6]                         0.0000
00007 GF[7]                         0.0000
00008 GF[8]                         0.0000
00009 GF[9]                         0.0000
00010 GF[10]                        0.0000


F1   F2   F3                   F7   F8
PgUp PgDn Fmt                  Main Diag
```

Display Format

Pressing **F3** toggles the display between float fixed (100.000) and scientific (1.000e+02) notation.

21.7 F3 Jog Menu

The Jog menu allows you to jog a stopped system. The following I/O register bits must be on before jogging an axis:

Register 1 - System Control		**Register 2, 3, 4, or 5 -Task Control**	
Bit 6	Pendant Live Man	Bit 1	Mode:! Manual
Bit 14	Pendant Enable		

```
             ROBOT JOG MENU
                    A
System: World
Method: Continuous/Slow
  0001: ABS[1]
                AXIS      WORLD      TAUGHT
01 Drive_1    12.643     47.500      20.300
02 Drive_2    95.215     18.300      54.200
03 Drive_3    63.609      5.500      16.000
04 Drive_4     0.960     36.800      10.000


00 Single     857.628

F1   F2          F4   F5   F6   F7   F8
Syst Meth        MvTo Axis Para Main Diag
```

F1 = System **F2 = Method** **F4 = Move To** **F5 = Axis**

F6 = Parameters **F7 = Main Screen** **F8 = Diagnostics**

Press **F1** to select either the Axis, Joint or World jog system. **F2** selects the jog method which can be continuos or incremental.

F4 is a "Move To" function that allows the user to enter a position in the TAUGHT column and move the specified axis to that point by pressing OK. Use the up and down arrow keys to move the cursor to the desired TAUGHT axis. This function is only available for coordinated axes.

F5 selects a single axis to jog.

F6 opens the *Edit Jog Parameters* screen which allows the user to adjust the percent distance and speed parameters, as well as, view the values set for each Task and Axis.

Jog Systems

Axis Jog Menu

The **Single Axis Jog** menu allows jogging a single, non-coordinated axis. Only the selected axis is affected. The BTC 06 display is continuously updated with the current position of the axis.

Press **A**– to jog in the negative direction.
Press **A**+ to jog in the positive direction.
(The teach pendant beeps at the beginning and end of motion.)

Coordinated Jogging

Press **X-** to jog in the negative X direction.
Press **X+** to jog in the positive X direction.

Press **Y-** to jog in the negative Y direction.
Press **Y+** to jog in the positive Y direction.

Press **Z-** to jog in the negative Z direction.
Press **Z+** to jog in the positive Z direction.

Joint Jog Menu

The **Joint** Jog menu allows jogging of individual axes with a joint number.

Robot World Jog Menu

The Robot **World** Jog menu allows jogging a coordinated or single axis for a task in World Cartesian Space. When jogging in world coordinates, motion will be generated parallel to the selected X, Y, or Z coordinate.

The pendant beeps at the beginning and end of motion. The display is continuously updated to display the current position (X, Y, Z) on each of the axes.

Jog Method

The following Jog Methods are available with the Teach Pendant:

Continuous/Fast	Continues to jog quickly until the button is released
Continuous/Slow	Continues to jog slowly until the button is released
Incremental/Large	Jogs a predetermined large increment and then stops.
Incremental/Small	Jogs a predetermined small increment and then stops.

Teaching Points

To teach the current position (during a coordinated jog) into the Absolute Point Table press TEACH. (Confirm each point by pressing the OK key.)

The table point number will flash indicating that point has been recorded in the table. The point number will automatically advance to the next point.

Jog Fine Adjustments

The jog speed and distance increments are set as a percentage of the Maximum Jog Increment and Maximum Jog Velocity parameters (T-0-0025 and T-0-0026).

Separate percents are used for FAST/SLOW and LARGE/SMALL jog settings in coordinated jog.

While in the Axis Jog or World Jog Menus, pressing F6 (PAR key) displays a screen that permits editing the FAST/SLOW and LARGE/SMALL jog percents.

21.8 F4 Control Menu

The Control menu allows the pendant to control the execution of a task.

When the Teach Pendant powers up, the Control Menu is the first menu displayed.

The Control Menu will provide the following information:

Title	Control Menu Title
Task Status	Current task operating status
Program Name	Name of the currently active program
Sequence	Name of the current sequence executing
Step	Name of the current step executing
Function	Name of the current function executing
Position Title	Axis Position Title (Joint, World, & Target)
Axis 1 Label (X)	Axis defined as axis 1
Axis 2 Label (Y)	Axis defined as axis 2
Axis 3 Label (Z)	Axis defined as axis 3
Axis 4 Label (A)	Axis defined as axis 4
Target Name	Point # and label for the current point executing
Function Keys	Function keys control machine operation
Operation Labels	Specify the machine operations

The control menu can run in one of three different modes. The following pages describe the operation of each mode and illustrate the different menu layouts.

Control Menu: Auto Run/Hold Mode

Text appears when the **F6** key, ***DBug*** is pressed.

Text disappears when the F6 key, ***Norm*** is pressed.

```
              CONTROL   MENU
INST: 00FC AXIS_WAIT
STAT: Move to return position
DIAG: Task Running         No Target
AXIS          WORLD        TARGET      STATUS
            1200.00       1200.00      Task A
              00.00        500.00      AUTO
              90.00         90.00      RUN
                                         99%


F1    F2    F3    F4    F5    F6    F7    F8
Auto  Go    Stop  I/O   Inst  Norm  Main  Diag
```

F1 - Mode Of Operation

If the *Teach Pendant Enabled Bit (Register 1 bit 14)* is high, pressing the **F1** key will change the mode of operation in the order shown below:

Manual: Jog ⇒ Auto: Step ⇒ Auto: Run/Hold ⇒ Auto: Step ⇒

F1 = Auto	**F2** = Go	**F3** = Stop	**F4** = I/O
F6 = Debug/Norm	**F7** = Main Menu	**F8** = Diagnostics	

Note: F2,**F3** and **F5** are dependent on the selected mode of operation.

When A*utomatic Mode* is selected by pressing the **F1** key, **F2** will display Go and **F3** will display Stop. By pressing the **F2** key, the active program will start executing instructions. By pressing the **F3** key, program execution will stop. If the **F2** key is pressed again, the program will continue.

To restart at the beginning of the program, the mode of operation must be changed to manual or step and then changed back to auto.

By pressing the **F4** key, the Register Menu will be displayed, allowing the operator to active I/O bits.

In Auto: Step mode **F5** is used to select the step method which can be one instruction, one Sequence Step, or one Sequence Function at a time.

When Debug is selected by pressing the **F6** key, the following text appears in the top half of the screen and the **F6** key text changes to Norm. Pressing **F6** (Normal) again removes this information.

CONTROL MENU	Screen name
INST: 00FC AXIS_WAIT	VisualMotion instruction being processed
STAT: Task Running	Task status
DIAG: Task Running	Diagnostic status

Control Menu: Auto Step Mode

Text appears when the **F6** key, ***DBug*** is pressed.

Text disappears when the F6 key, ***Norm*** is pressed.

```
              CONTROL   MENU
INST: 0194 SET
STAT: In return position
DIAG: Instruction Sin     No Target
AXIS          WORLD        TARGET     STATUS
            1200.00       1200.00     Task A
              00.00        500.00     AUTO
              90.00         90.00     RUN
                                        99%



F1   F2   F3   F4   F5   F6   F7   F8
Step Go   Stop I/O  Inst Norm Main Diag
```

F1 = Step **F2** = Go **F3** = Stop **F4** = I/O

F5 = Instruction/Sequence **F6** = DeBug/Norm

F7 = Main Menu **F8** = Diagnostics

When the Automatic Step Mode is selected by pressing the **F1** key, **F2** will display GO and **F3** will display STOP. Every time the **F2**-GO key is pressed, the program will be sequentially executed one step at a time. The steps can be program instructions, Sequencer steps or Sequencer functions. Pressing the **F5** key selects the step mode that the program will follow:

Instruction	-	INST
Sequence/Steps	-	SEQ/STEP
Sequence/Function	-	SEQ/FUNC

When **F5** (INST) is selected the program will execute only one instruction every time the **F2**-GO key is pressed. When SEQ/STEP is selected the program will execute all the functions within one Sequencer step, one at a time. When SEQ/FUNC is selected the program will execute each Sequencer function, one at a time.

The **F3**-STOP key can be used to immediately halt the execution of the program within a step. If the **F2**-GO key is pressed again, the step will continue to run.

By pressing the **F4** key, the Register Menu will be displayed, allowing the operator to active I/O bits.

Control Menu: Manual Mode

Text appears when the **F6** key, ***DBug*** is pressed.

Text disappears when the F6 key, ***Norm*** is pressed.

```
                CONTROL   MENU
INST: 0194 SET
STAT: In return position
DIAG: Manual mode              No Target
AXIS          WORLD            TARGET      STATUS
            1200.00           1200.00      Task A
              00.00            500.00      AUTO
              90.00             90.00      RUN
                                               99%


F1   F2   F3   F4   F5   F6   F7   F8
Manu Go   Stop I/O  Inst Norm Main Diag
```

F1 = Manual **F2 =** Jog **F4 =** I/O **F6** = DeBug/Norm

F7 = Main Menu **F8 =** Diagnostics

By pressing the **F2** key, the Jog Menu will be displayed, allowing the operator to jog and teach each axis.

By pressing the **F4** key, the Register Menu will be displayed, allowing the operator to active I/O bits.

21.9 F5 Register I/O Menu

The **F4** I/O key is provided on every operator interface control screen. The operator will have the ability to view and edit the register bits that the machine builder selects. When the **F4** key is pressed, the register menu will be displayed. The first register displayed is set in control parameter C-0-0805 *Start of User Accessible Registers on Pendant*. Parameter C-0806 defines the *End of User Accessible Registers on Pendant*. The operator can only edit registers within the range of these two parameters.

For example, parameter C-0805 = register 100, which is labeled End_Of_Arm_Tool_1. When the **F4** key is pressed, the first screen displayed is register 100, along with the register label. Bits 1 through 8 are displayed, along with the bit labels and current state (ON/OFF). This screen is only updated when any of the bits change states.

```
              REGISTER MENU

REGISTER: 0100  End_Of_Arm_Tool_1

BITS: Forward                     01 OFF
      Reverse                     02 OFF
      C_Axis_Vertical             03 ON
      C_Axis_Horizontal           04 OFF
      A_Axis_Forward              05 OFF
      A_Axis_Retracted            06 OFF
      Bit_07                      07 OFF
      Bit_08                      08 OFF

      Down Arrow For Bits 09 - 16
F1    F2    F3    F4    F5    F6    F7    F8
01    02    03    04    05    06    07    08
```

The first page of the register menu will display bits 1 through 8. By pressing the down arrow key, bits 9 through 16 will be displayed. Pressing the up arrow key will return to bits 1 through 8.

The page up  and page down  keys move the cursor to the next or previous register available within the limits of card parameters C-0-0805 and C-0-0806.

To jump to a register number:

1. Press the edit key
2. Enter the register number
3. Press the OK key

```
                  REGISTER MENU

REGISTER: 0100   End_Of_Arm_Tool_1

BITS: Bit_09                     01 OFF
      Bit_10                     02 OFF
      Bit_11                     03 ON
      Bit_12                     04 OFF
      Bit_13                     05 OFF
      Bit_14                     06 OFF
      Bit_15                     07 OFF
      Bit_16                     08 OFF

      Down Arrow For Bits 01 - 08
F1    F2    F3    F4    F5    F6    F7    F8
01    02    03    04    05    06    07    08
```

The function keys **F1** through **F8** will allow the operator to toggle the state (ON/OFF) of bits 1 through 8 (first page displayed) or bits 9 through 16 (second page displayed).

Note: If an operator needs to change a bit in a register outside the range set by parameters C-0-0805 and C-0-0806, a password will have to be entered or the pendant level protection bits will have to be adjusted. See the Security Menu description.

21.10 F6 Parameter Menu

The Parameter menu allows selection of screens for editing the system, task, axis, and drive parameters.

```
                  PARAMETER MENU

                    F1  CARD
                    F2  AXIS
                    F3  TASK
                    F4  DRIVE



                                    F7    F8
                                    Main  Diag
```

Selecting **F1-F4** will open one of the following Parameter screens.

F1 - Card Parameter Screen

```
          CARD PARAMETER MENU
C-c-ppppp nnnnn                            vvvv
C-0-00001 Language Selection                  1
…
…
…
…
…
…
…
…
C-0-00042 World Large Increment              50


F1   F2   F3   F4              F7   F8
Home End  PgUp PgDn            Main Diag
```

C-c-pppp **nnnn** **vvvv**

vvvv = parameter value

nnnn = parameter name

pppp = parameter number

c = control number

C = system parameter indicator

For changing of Card parameter numbers and editing of values, refer to **F2 - Axis Parameter Screen** on page 21-24.

F2 - Axis Parameter Screen

```
          AXIS PARAMETER MENU
Aaa-ppppp nnnnn                            vvvv
A01-00001 Language Selection                  1
…
…
…
…
…
…
…
…
A01-000021 Maximum Acceleration             200


F1   F2   F3   F4              F7   F8
Home End  PgUp PgDn            Main Diag
```

Aaa-pppp **nnnn** **vvvv**

vvvv = parameter value

nnnn = parameter name

pppp = parameter number

aa = axis number

A = axis parameter indicator

When the operator first enters this screen, the cursor will be flashing on the **aa** axis number. This can also be performed by pressing the **F1 Home** key.

To change the axis number:

1. Press the Edit key
2. Enter the new axis number
3. Press the OK key

When done, all of the axis parameter number will be modified to display the new axis number.

To move the cursor to the parameter number (**pppp)** while the cursor is on the axis number, press the right arrow key. Pressing the **F1 Home** key will return the cursor to the axis number.

To jump to a given parameter number:

1. Press the Edit key
2. Enter the parameter number
3. Press the OK key

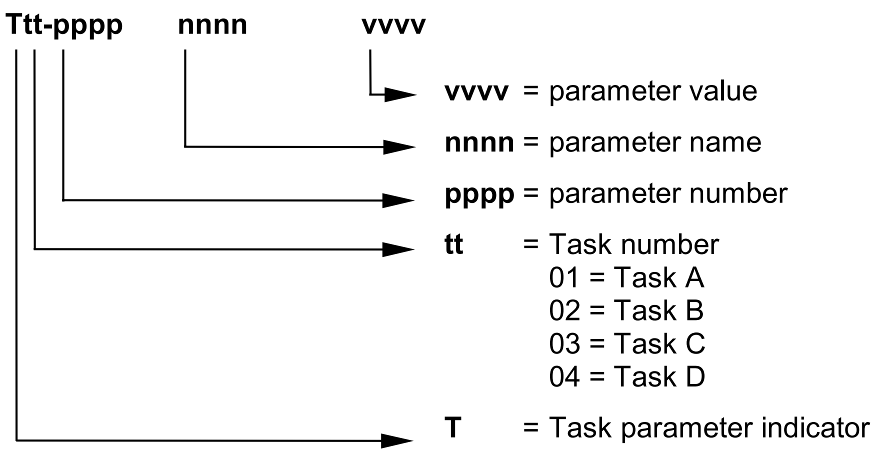
When done, the cursor will jump to the specified parameter number.

To modify the value for a given parameter, press the **F2 End** key to jump to the value field. Press the Edit key, enter the new value and press OK.

F3 - Task Parameter Screen

```
          TASK PARAMETER MENU
Ttt-ppppp nnnnn                        vvvv
T01-00001 Task Motion Type                1
…
…
…
…
…
…
…
…
T01-00022 Maximum Deceleration         200

F1   F2   F3   F4             F7   F8
Home End  PgUp PgDn           Main Diag
```



For changing of Task parameter numbers and editing of values, refer to **F2 - Axis Parameter Screen** on page 21-24.

F4 - Drive Parameter Screen

```
             DRIVE PARAMETER MENU
Ddd-ppppp nnnnn                         vvvv
S01-00001 NC Cycle time (TN)            2000
…
…
…
…
…
…
…
…
S01-000010 Length of master               20


F1    F2    F3    F4                F7    F8
Home  End   PgUp  PgDn              Main  Diag
```

Ddd-pppp nnnn vvvv

vvvv = parameter value

nnnn = parameter name

pppp = parameter number

dd = Drive number

D = Drive parameter indicator
S = SERCOS
P = Product Specific

For changing of Drive parameter numbers and editing of values, refer to **F2 - Axis Parameter Screen** on page 21-24.

This Drive Parameter Menu screen contains S (SERCOS) parameters and P (Product Specific) parameters. Instead of paging down to view P parameters, use the following steps to quickly reach them.

1. Position the cursor over the parameter number
2. Press the Edit key
3. Enter the last SERCOS number (412) and press OK
4. S-0-0412 will appear at the bottom of the list, press **F4** page down to display the first P parameter (P-0-0004).
5. To return back to the start of the SERCOS parameters, page up and position the cursor on a SERCOS parameter.
6. Press the Edit key
7. Enter the first SERCOS number (1) and press OK.

21.11 F6 Security Menu

The Security Menu allows the Teach Pendant manager to assign a protection level code between 0 and 2 for each menu. Different access codes can then be set for various users to provide customized security for system data.

Note: The operation of the BTC 06 Security menu is related to the setting in control parameter C-0-0801.

```
             SECURITY MENU

001 CLC PROGRAM MENU                  1
002 CLC TABLE EDIT MENU               1
003 ROBOT JOG MENU                    1
004 CONTROL MENU                      0
005 REGISTER MENU                     2
006 SECURITY MENU                    -1
007 CARD PARAMETER MENU               1
008 AXIS PARAMETER MENU               1
009 TASK PARAMETER MENU               1
010 DRIVE PARAMETER MENU              1
011 CLC MAIN MENU                     0
                               F7   F8
                               Main Diag
```

To alter a menu protection level, place the cursor over the protection level field and press the EDIT button. Key-in the appropriate code (0, 1 or 2) and press OK. The Security Level Menu has a default of -1 to allow initial access to all users.

The user access status for each menu depends on the menu protection level outlined above and the users access code, which is determined by the System Control Register 1, Bits 15 and 16. The access code has to be greater than the menu protection level to allow the user to view and edit a menu. If the levels are the same, the user can only view the menu. A menu with a protection level that is higher than the security level cannot be accessed by a user. The following table lists the level combinations, which determine user access privileges.

Bit 15 Status	Bit 16 Status	Access Code (Bit Resultant)	Protection Level (Preset)	Net Access Status
0	0	0	0 1,2	View Only No Access
1	0	1	0 1 2	View/Edit View Only No Access
0	1	2	0,1 2	View/Edit View Only
1	1	3	0-2	View/Edit

21.12 F8 Diagnostics Menu

When the **F8** key is pressed from any of the Operator Interface Control Menus, the Diagnostics Menu is displayed. The diagnostics menu displays the current Card, Tasks, Axis and Drive status. The diagnostics screen updates continuously. When first entering this menu, by default, Axs=1, Drv=1 and the cursor is positioned on the control number.

```
               DIAGNOSTICS MENU

Crd: 03 007 Program Running: AB
            Extended Diagnostic Message

Tsk:AA TASK RUNNING
Tsk:BB TASK RUNNING
Tsk:CC MANUAL MODE
Tsk:DD MANUAL MODE

Axs:01 No Axis Message

Drv: 02 303 Position Mode Encoder 1 / lag

F1   F2   F3   F4              F7
Num+ Num- Rest Log             Main
```

Positioning The Cursor

The up/down arrow keys will position the cursor on the menu item the user may wish to edit.

NOTE: *The Teach Pendant cannot edit the card number or task.*

Cursor Positioned Axis and Drive Number

By pressing the (F1 Num+) or (F2 Num-), the Axis number will increment up or down, respectively.

By pressing the Edit key, the operator can enter the desired Axis number and press the OK key to accept.

By pressing the (F3 Reset) key, the information on the screen is refreshed.

By pressing the (F4 Log) key, the screen will display a list of error that contain the following details:

- Log number
- Date and Time
- Error code

For a complete listing of Diagnostics, refer to the *VisualMotion 11 Trouble Shooting Guide*, Monitoring and Diagnostics.

21.13 Error Screen

If an error is detected during operation, the pendant automatically enters the Error screen and displays a message about the error condition.

If the *BTC 06 enable bit (Register 1, bit 14)* is on and an error occurs, the BTC 06 will force all tasks into manual mode.

Pressing escape after an error occurs will display the Diagnostic Menu.

```
            DIAGNOSTICS MENU
Crd:03 420 Drive 6 Shutdown Error

Tsk:AA MANUAL MODE
Tsk:BB MANUAL MODE
Tsk:CC MANUAL MODE
Tsk:DD MANUAL MODE

Axs:01 No Axis Message

Drv:04 028 Excessive Deviation

F1   F2   F3   F4              F7
Num+ Num- Rest Log             Main
```

F3 - Reset

A basic “Shutdown” error can be cleared by pressing **F3**. If the error is a configuration or hardware error, the source of that error must first be corrected before it can be cleared by the pendant.

Index

A

Rexroth
Bosch Group

Rexroth
Bosch Group

Rexroth
Bosch Group

D

P

Rexroth
Bosch Group

W

X

Z

Rexroth
Bosch Group

Service & Support

Helpdesk

Unser Kundendienst-Helpdesk im Hauptwerk Lohr am Main steht Ihnen mit Rat und Tat zur Seite. Sie erreichen uns

Our service helpdesk at our headquarters in Lohr am Main, Germany can assist you in all kinds of inquiries. Contact us

telefonisch - by phone: **+49 (0) 9352 40 50 60**
über Service Call Entry Center Mo-Fr 07:00-18:00 Central European Time
- via Service Call Entry Center Mo-Fr 7:00 am - 6:00 pm CET

per Fax - by fax: **+49 (0) 9352 40 49 41**

per e-Mail - by e-mail: **service.svc@boschrexroth.de**

Service-Hotline

Außerhalb der Helpdesk-Zeiten ist der Service Deutschland direkt ansprechbar unter

After helpdesk hours, contact the German service experts directly at

+49 (0) 171 333 88 26

oder - or **+49 (0) 172 660 04 06**

Hotline-Rufnummern anderer Länder entnehmen Sie bitte den Adressen in den jeweiligen Regionen.

Hotline numbers of other countries to be seen in the addresses of each region.

Internet

Unter **www.boschrexroth.com** finden Sie ergänzende Hinweise zu Service, Reparatur und Training sowie die **aktuellen** Adressen *) unserer auf den folgenden Seiten aufgeführten Vertriebs- und Servicebüros.

- ☐ Verkaufsniederlassungen
- ☐ Niederlassungen mit Kundendienst

Außerhalb Deutschlands nehmen Sie bitte zuerst Kontakt mit unserem für Sie nächstgelegenen Ansprechpartner auf.

*) Die Angaben in der vorliegenden Dokumentation können seit Drucklegung überholt sein.

At **www.boschrexroth.com** you can find additional notes about service, repairs and training. The current addresses *) for our sales and service locations around the world are on the following pages.

- ☐ Sales agencies
- ☐ Sales agencies providing service

Please contact our sales / service office in your area first.

*) Data in the present documentation may have become obsolete since printing.

Vor der Kontaktaufnahme... - Before contacting us...

Wir können Ihnen schnell und effizient helfen wenn Sie folgende Informationen bereithalten:

Detaillierte Beschreibung der Störung und der Umstände.

Angaben auf dem Typenschild der betreffenden Produkte, insbesondere Typenschlüssel und Seriennummern.

Tel.-/Faxnummern und e-Mail-Adresse, unter denen Sie für Rückfragen zu erreichen sind.

For quick and efficient help, please have the following information ready:

Detailed description of the failure and circumstances.

Information on the nameplate of the affected products, especially typecodes and serial numbers.

Your phone/fax numbers and e-mail address, so we can contact you in case of questions.

Kundenbetreuungsstellen - Sales & Service Facilities

Deutschland – Germany

vom Ausland: (0) nach Landeskennziffer weglassen!
from abroad: don't dial (0) after country code!

Vertriebsgebiet Mitte Germany Centre Bosch Rexroth Electrice Drives and Controls GmbH Bgm.-Dr.-Nebel-Str. 2 / Postf. 1357 97816 Lohr am Main / 97803 Lohr **Kompetenz-Zentrum Europa** Tel.: +49 (0)9352 40-0 Fax: +49 (0)9352 40-4885	SERVICE AUTOMATION CALL ENTRY CENTER **Helpdesk** **MO – FR** **von 07:00 - 18:00 Uhr** **from 7 am – 6 pm** **Tel. +49 (0) 9352 40 50 60** **Fax +49 (0) 9352 40 49 41** service.svc@boschrexroth.de	SERVICE AUTOMATION **HOTLINE 24 / 7 / 365** **außerhalb der Helpdesk-Zeit** **out of helpdesk hours** **Tel.: +49 (0)172 660 04 06** oder / or **Tel.: +49 (0)171 333 88 26**	SERVICE AUTOMATION **ERSATZTEILE / SPARES** verlängerte Ansprechzeit - extended office time - nur an Werktagen - only on working days - von 07:00 - 18:00 Uhr - from 7 am - 6 pm - **Tel. +49 (0) 9352 40 42 22**
Vertriebsgebiet Süd Germany South Bosch Rexroth AG Landshuter Allee 8-10 80637 München Tel.: +49 (0)89 127 14-0 Fax: +49 (0)89 127 14-490	Vertriebsgebiet West Germany West Bosch Rexroth AG Regionalzentrum West Borsigstrasse 15 40880 Ratingen Tel.: +49 (0)2102 409-0 Fax: +49 (0)2102 409-406 +49 (0)2102 409-430	Gebiet Südwest Germany South-West Bosch Rexroth AG Service-Regionalzentrum Süd-West Siemensstr. 1 70736 Fellbach Tel.: +49 (0)711 51046–0 Fax: +49 (0)711 51046–248	
Vertriebsgebiet Nord Germany North Bosch Rexroth AG Walsroder Str. 93 30853 Langenhagen Tel.: +49 (0) 511 72 66 57-0 Service: +49 (0) 511 72 66 57-256 Fax: +49 (0) 511 72 66 57-93 Service: +49 (0) 511 72 66 57-783	Vertriebsgebiet Mitte Germany Centre Bosch Rexroth AG Regionalzentrum Mitte Waldecker Straße 13 64546 Mörfelden-Walldorf Tel.: +49 (0) 61 05 702-3 Fax: +49 (0) 61 05 702-444	Vertriebsgebiet Ost Germany East Bosch Rexroth AG Beckerstraße 31 09120 Chemnitz Tel.: +49 (0)371 35 55-0 Fax: +49 (0)371 35 55-333	Vertriebsgebiet Ost Germany East Bosch Rexroth AG Regionalzentrum Ost Walter-Köhn-Str. 4d 04356 Leipzig Tel.: +49 (0)341 25 61-0 Fax: +49 (0)341 25 61-111

Europa (West) - Europe (West)

vom Ausland: (0) nach Landeskennziffer weglassen, Italien: 0 nach Landeskennziffer mitwählen
from abroad: don't dial (0) after country code, Italy: dial 0 after country code

Austria - Österreich Bosch Rexroth GmbH Electric Drives & Controls Stachegasse 13 1120 Wien Tel.: +43 (0) 1 985 25 40 Fax: +43 (0) 1 985 25 40-1459	Austria – Österreich Bosch Rexroth GmbH Electric Drives & Controls Industriepark 18 4061 Pasching Tel.: +43 (0)7221 605-0 Fax: +43 (0)7221 605-1220	Belgium - Belgien Bosch Rexroth NV/SA Henri Genessestraat 1 1070 Bruxelles Tel: +32 (0) 2 451 26 08 Fax: +32 (0) 2 451 27 90 info@boschrexroth.be service@boschrexroth.be	Denmark - Dänemark BEC A/S Zinkvej 6 8900 Randers Tel.: +45 87 11 90 60 Fax: +45 87 11 90 61
Denmark - Dänemark Bosch Rexroth A/S Engelsholmvej 26 8900 Randers Tel.: +45 36 77 44 66 Fax: +45 70 10 03 20 tj@boschrexroth.dk	Great Britain – Großbritannien Bosch Rexroth Ltd. Electric Drives & Controls Broadway Lane, South Cerney Cirencester, Glos GL7 5UH Tel.: +44 (0)1285 863-000 Fax: +44 (0)1285 863-030 sales@boschrexroth.co.uk service@boschrexroth.co.uk	Finland - Finnland Bosch Rexroth Oy Electric Drives & Controls Ansatie 6 017 40 Vantaa Tel.: +358 (0)9 84 91-11 Fax: +358 (0)9 84 91-13 60	France - Frankreich Bosch Rexroth SAS Electric Drives & Controls Avenue de la Trentaine (BP. 74) 77503 Chelles Cedex Tel.: +33 (0)164 72-63 22 Fax: +33 (0)164 72-63 20 **Hotline: +33 (0)608 33 43 28**
France - Frankreich Bosch Rexroth SAS Electric Drives & Controls ZI de Thibaud, 20 bd. Thibaud (BP. 1751) 31084 Toulouse Tel.: +33 (0)5 61 43 61 87 Fax: +33 (0)5 61 43 94 12	France – Frankreich Bosch Rexroth SAS Electric Drives & Controls 91, Bd. Irène Joliot-Curie 69634 Vénissieux – Cedex Tel.: +33 (0)4 78 78 53 65 Fax: +33 (0)4 78 78 53 62	France – Frankreich **Tightening & Press-fit:** Globe Techniques Nouvelles 143, Av. du Général de Gaulle 92252 La Garenne Colombes Tel.: +33 (0)1 41 19 33 33	Italy - Italien Bosch Rexroth S.p.A. Strada Statale Padana Superiore 11, no. 41 20063 Cernusco S/N.MI **Hotline: +39 02 92 365 563** Tel.: +39 02 92 365 1 Service: +39 02 92 365 300 Fax: +39 02 92 365 500 Service: +39 02 92 365 516
Italy - Italien Bosch Rexroth S.p.A. Via Paolo Veronesi, 250 10148 Torino Tel.: +39 011 224 88 11 Fax: +39 011 224 88 30	Italy - Italien Bosch Rexroth S.p.A. Via Mascia, 1 80053 Castellamare di Stabia NA Tel.: +39 081 8 71 57 00 Fax: +39 081 8 71 68 85	Italy - Italien Bosch Rexroth S.p.A. Via del Progresso, 16 (Zona Ind.) 35020 Padova Tel.: +39 049 8 70 13 70 Fax: +39 049 8 70 13 77	Italy - Italien Bosch Rexroth S.p.A. Via Isonzo, 61 40033 Casalecchio di Reno (Bo) Tel.: +39 051 29 86 430 Fax: +39 051 29 86 490
Italy - Italien **Tightening & Press-fit:** TEMA S.p.A. Automazione Via Juker, 28 20025 Legnano Tel.: +39 0 331 4671	Netherlands – Niederlande/Holland Bosch Rexroth B.V. Kruisbroeksestraat 1 (P.O. Box 32) 5281 RV Boxtel Tel.: +31 (0) 411 65 16 40 Fax: +31 (0) 411 65 14 83 www.boschrexroth.nl	Netherlands - Niederlande/Holland Bosch Rexroth Services B.V. Technical Services Kruisbroeksestraat 1 (P.O. Box 32) 5281 RV Boxtel Tel.: +31 (0) 411 65 19 51 Fax: +31 (0) 411 67 78 14 **Hotline: +31 (0) 411 65 19 51** services@boschrexroth.nl	Norway - Norwegen Bosch Rexroth AS Electric Drives & Controls Berghagan 1 or: Box 3007 1405 Ski-Langhus 1402 Ski Tel.: +47 64 86 41 00 Fax: +47 64 86 90 62 **Hotline: +47 64 86 94 82** jul.ruud@rexroth.no
Spain – Spanien Goimendi Automation S.L. Parque Empresarial Zuatzu C/ Francisco Grandmontagne no.2 20018 San Sebastian Tel.: +34 9 43 31 84 21 - service: +34 9 43 31 84 56 Fax: +34 9 43 31 84 27 - service: +34 9 43 31 84 60 sat.indramat@goimendi.es	Spain - Spanien Bosch Rexroth S.A. Electric Drives & Controls Centro Industrial Santiga Obradors 14-16 08130 Santa Perpetua de Mogoda Barcelona Tel.: +34 9 37 47 94-00 Fax: +34 9 37 47 94-01	Spain - Spanien Bosch Rexroth S.A. Electric Drives & Controls c/ Almazara, 9 28760 Tres Cantos (Madrid) Tel.: +34 91 806 24 79 Fax: +34 91 806 24 72 fernando.bariego@boschrexroth.es	Sweden - Schweden Bosch Rexroth AB Electric Drives & Controls - Varuvägen 7 (Service: Konsumentvägen 4, Älfsjö) 125 81 Stockholm Tel.: +46 (0) 8 727 92 00 Fax: +46 (0) 8 647 32 77
Sweden - Schweden Bosch Rexroth AB Electric Drives & Controls Ekvändan 7 254 67 Helsingborg Tel.: +46 (0) 4 238 88 -50 Fax: +46 (0) 4 238 88 -74	Switzerland East - Schweiz Ost Bosch Rexroth Schweiz AG Electric Drives & Controls Hemrietstrasse 2 8863 Buttikon Tel. +41 (0) 55 46 46 111 Fax +41 (0) 55 46 46 222	Switzerland West - Schweiz West Bosch Rexroth Suisse SA Av. Général Guisan 26 1800 Vevey 1 Tel.: +41 (0)21 632 84 20 Fax: +41 (0)21 632 84 21	

Europa (Ost) - Europe (East)

vom Ausland: (0) nach Landeskennziffer weglassen
from abroad: don't dial (0) after country code

Czech Republic - Tschechien Bosch -Rexroth, spol.s.r.o. Hviezdoslavova 5 627 00 Brno Tel.: +420 (0)5 48 126 358 Fax: +420 (0)5 48 126 112	Czech Republic - Tschechien **Tightening & Press-fit:** Bosch -Rexroth, spol.s.r.o. Stetkova 18 140 68 Praha 4 Tel.: +420 (0)241 406 675	Hungary - Ungarn Bosch Rexroth Kft. Angol utca 34 1149 Budapest Tel.: +36 (1) 422 3200 Fax: +36 (1) 422 3201	Poland – Polen Bosch Rexroth Sp.zo.o. ul. Staszica 1 05-800 Pruszków Tel.: +48 (0) 22 738 18 00 – service: +48 (0) 22 738 18 46 Fax: +48 (0) 22 758 87 35 – service: +48 (0) 22 738 18 42
Poland – Polen Bosch Rexroth Sp.zo.o. Biuro Poznan ul. Dabrowskiego 81/85 60-529 Poznan Tel.: +48 061 847 64 62 /-63 Fax: +48 061 847 64 02	Romania - Rumänien East Electric S.R.L. Bdul Basarabia no.250, sector 3 73429 Bucuresti Tel./Fax:: +40 (0)21 255 35 07 +40 (0)21 255 77 13 Fax: +40 (0)21 725 61 21 eastel@rdsnet.ro	Romania - Rumänien Bosch Rexroth Sp.zo.o. Str. Drobety nr. 4-10, app. 14 70258 Bucuresti, Sector 2 Tel.: +40 (0)1 210 48 25 +40 (0)1 210 29 50 Fax: +40 (0)1 210 29 52	Russia - Russland Bosch Rexroth OOO Tschschjolkowskoje Chaussee 100 105523 Moskau Tel.: +7-495-783 30 60 + 64 Fax: +7-495 783 30 68 + 69 brcschrexroth.ru
Turkey - Türkei Bosch Rexroth Otomasyon San & Tic. A..S. Fevzi Cakmak Cad No. 3 34295 Sefaköy Istanbul Tel.: +90 212 411-13 00 Fax: +90 212 411-13 17 www.boschrexroth.com.tr	Turkey - Türkei Servo Kontrol Ltd. Sti. Perpa Ticaret Merkezi B Blok Kat: 11 No: 1609 80270 Okmeydani-Istanbul Tel: +90 212 320 30 80 Fax: +90 212 320 30 81 remzi.sali@servokontrol.com www.servokontrol.com	Slowakia - Slowakei **Tightening & Press-fit:** MTS, spol .s.r.o. 02755 Kriva 53 Tel.: +421 43 5819 161	Slowenia - Slowenien DOMEL Otoki 21 64 228 Zelezniki Tel.: +386 5 5117 152 Fax: +386 5 5117 225 brane.ozebek@domel.si

Australien, Süd-Afrika - Australia, South Africa

Australia - Australien AIMS - Australian Industrial Machinery Services Pty. Ltd. 28 Westside Drive Laverton North Vic 3026 Melbourne Tel.: +61 3 93 14 3321 Fax: +61 3 93 14 3329 **Hotlines: +61 3 93 14 3321** **+61 4 19 369 195** enquires@aimservices.com.au	Australia - Australien Bosch Rexroth Pty. Ltd. No. 7, Endeavour Way Braeside Victoria, 31 95 Melbourne Tel.: +61 3 95 80 39 33 Fax: +61 3 95 80 17 33 mel@rexroth.com.au	South Africa - Südafrika TECTRA Automation (Pty) Ltd. 100 Newton Road, Meadowdale Edenvale 1609 Tel.: +27 11 971 94 00 Fax: +27 11 971 94 40 **Hotline: +27 82 903 29 23** georgv@tectra.co.za	South Africa - Südafrika **Tightening & Press-fit:** Jendamark Automation 76A York Road, North End 6000 Port Elizabeth Tel.: +27 41 391 4735

Asien - Asia (incl. Pacific Rim)

China
Shanghai Bosch Rexroth
Hydraulics & Automation Ltd.
No.122, Fu Te Dong Yi Road
Waigaoqiao, Free Trade Zone
Shanghai 200131 - P.R.China

Tel.: +86 21 58 66 30 30
Fax: +86 21 58 66 55 23
brcservice@boschrexroth.com.cn

China
Shanghai Bosch Rexroth
Hydraulics & Automation Ltd.
4/f, Marine Tower
No.1, Pudong Avenue
Shanghai 200120 - P.R.China

Tel: +86 21 68 86 15 88
Fax: +86 21 68 86 05 99
+86 21 58 40 65 77
y.wu@boschrexroth.com.cn

China
Bosch Rexroth (China) Ltd.
Satellite Service Office Changchun
Rm. 1910, Guangming Building
No.336 Xi'an Rd., Chao Yang Distr.
Changchun 130061 - P.R.China

Tel.+Fax: +86 431 898 1129
Mobile: +86 139 431 92 659
shuhong.wang@boschrexroth.com.cn

China
Bosch Rexroth (China) Ltd.
Satellite Service Office Wuhan
No. 22, Pinglanju, Milanyuan, Golden Harbor
No. 236 Longyang Avenue
Economic & Technology Development Zone
Wuhan 430056 - P.R.China

Tel.+Fax: +86 27 84 23 23 92
Mobile: +86 139 71 58 89 67
ym.zhu@boschrexroth.com.cn

China
Bosch Rexroth (China) Ltd.
Beijing Representative Office
Xi San Qi Dong, De Sheng Mei Wai
Hai Dian District
Beijing 100096, P.R.China
Tel.: +86 10 82 91 22 29
Fax: +86 10 82 91 81 09
yan.zhang@boschrexroth.com.cn

China
Bosch Rexroth (China) Ltd.
Guangzhou Repres. Office
Room 3710-3716, Metro Plaza,
Tian He District, 183 Tian He Bei Rd
Guangzhou 510075, P.R.China
Tel.: +86 20 87 55 00 30
+86 20 87 55 00 11
Fax: +86 20 87 55 23 87
andrew.wang@boschrexroth.com.cn

China
Bosch Rexroth (China) Ltd.
Dalian Representative Office
Room 2005,Pearl River Int. Building
No.99 Xin Kai Rd., Xi Gang District
Dalian, 116011, P.R.China

Tel.: +86 411 83 68 26 02
Fax: +86 411 83 68 27 02
jason.tan@boschrexroth.com.cn

China
Tightening & Press-fit:
C. Melchers GmbH & Co
Shanghai Representation
13 Floor Est Ocean Centre
No.588 Yanan Rd. East
65 Yanan Rd. West
Shanghai 200001
Tel.: +86 21 63 52 88 48
Fax: +86 21 63 51 31 38
shermanxia@sh.melchers.com.cn

Hongkong
Bosch Rexroth (China) Ltd.
6th Floor,
Yeung Yiu Chung No.6 Ind Bldg.
19 Cheung Shun Street
Cheung Sha Wan,
Kowloon, Hongkong
Tel.: +852 27 86 46 32
Fax: +852 27 42 60 57
Paul.li@boschrexroth.com.cn

India - Indien
Bosch Rexroth (India) Ltd.
Electric Drives & Controls
Plot. No.96, Phase III
Peenya Industrial Area
Bangalore – 560058
Tel.: +91 80 51 17 0-211...-218
Fax: +91 80 83 94 345
+91 80 83 97 374
mohanvelu.t@boschrexroth.co.in

India - Indien
Bosch Rexroth (India) Ltd.
Electric Drives & Controls
Advance House, II Floor
Ark Industrial Compound
Narol Naka, Makwana Road
Andheri (East), Mumbai - 400 059
Tel.: +91 22 28 56 32 90
+91 22 28 56 33 18
Fax: +91 22 28 56 32 93
singh.op@boschrexroth.co.in

India - Indien
Tightening & Press-fit:

MICO
Hosur Road Adugodi
560 030 Bangalore Karnataki
Tel.: +91 80 22 99 28 86

India - Indien
Bosch Rexroth (India) Ltd.
S-10, Green Park Extension
New Delhi – 110016

Tel.: +91 11 26 56 65 25
+91 11 26 56 65 27
Fax: +91 11 26 56 68 87
koul.rp@boschrexroth.co.in

Indonesia - Indonesien
PT. Bosch Rexroth
Building # 202, Cilandak Commercial Estate
Jl. Cilandak KKO, Jakarta 12560
Tel.: +62 21 7891169 (5 lines)
Fax: +62 21 7891170 - 71
rudy.karimun@boschrexroth.co.id

Japan
Bosch Rexroth Automation Corp.
Service Center Japan
Yutakagaoka 1810, Meito-ku,
NAGOYA 465-0035, Japan
Tel.: +81 52 777 88 41
+81 52 777 88 53
+81 52 777 88 79
Fax: +81 52 777 89 01

Japan
Bosch Rexroth Automation Corp.
Electric Drives & Controls
2F, I.R. Building
Nakamachidai 4-26-44, Tsuzuki-ku
YOKOHAMA 224-0041, Japan
Tel.: +81 45 942 72 10
Fax: +81 45 942 03 41

Korea
Bosch Rexroth-Korea Ltd.
Electric Drives & Controls
1515-14 Dadae-Dong, Saha-gu
Pusan Metropolitan City, 604-050

Tel.: +82 51 26 00 741
Fax: +82 51 26 00 747
eunkyong.kim@boschrexroth.co.kr

Korea
Bosch Rexroth-Korea Ltd.
Electric Drives and Controls
Bongwoo Bldg. 7FL, 31-7, 1Ga
Jangchoong-dong, Jung-gu
Seoul, 100-391

Tel.: +82 234 061 813
Fax: +82 222 641 295

Korea
Bosch Rexroth-Korea Ltd.
Electric Drives & Controls
1515-14 Dadae-Dong, Saha-gu
Ulsan, 680-010

Tel.: +82 52 256-0734
Fax: +82 52 256-0738
keonhyun.jeong@boschrexroth.co.kr

Korea
Tightening & Press-fit:
KVT Co., Ltd.
901, Daeryung Techno Town 8
481-11 Gasan-Dong
Geumcheon-Gu
Seoul, 153-775
Tel.: +82 2 2163 0231 9

Malaysia
Bosch Rexroth Sdn.Bhd.
11, Jalan U8/82, Seksyen U8
40150 Shah Alam
Selangor, Malaysia

Tel.: +60 3 78 44 80 00
Fax: +60 3 78 45 48 00
hhlim@boschrexroth.com.my
rexroth1@tm.net.my

Singapore - Singapur
Bosch Rexroth Pte Ltd
15D Tuas Road
Singapore 638520

Tel.: +65 68 61 87 33
Fax: +65 68 61 18 25
lai.ts@boschrexroth.com.sg

Taiwan
Bosch Rexroth Co., Ltd.
Taichung Industrial Area
No.19, 38 Road
Taichung, Taiwan 407, R.O.C.
Tel : +886 - 4 -235 08 383
Fax: +886 - 4 -235 08 586
jim.lin@boschrexroth.com.tw
david.lai@boschrexroth.com.tw

Taiwan
Bosch Rexroth Co., Ltd.
Tainan Branch
No. 17, Alley 24, Lane 737
Chung Cheng N.Rd. Yungkang
Tainan Hsien, Taiwan, R.O.C.
Tel : +886 - 6 –253 6565
Fax: +886 - 6 –253 4754
charlie.chen@boschrexroth.com.tw

Thailand
NC Advance Technology Co. Ltd.
59/76 Moo 9
Ramintra road 34
Tharang, Bangkhen,
Bangkok 10230

Tel.: +66 2 943 70 62
+66 2 943 71 21
Fax: +66 2 509 23 62
Hotline +66 1 984 61 52
sonkawin@hotmail.com

Nordamerika – North America

USA Headquarters - Hauptniederlassung Bosch Rexroth Corporation Electric Drives & Controls 5150 Prairie Stone Parkway Hoffman Estates, IL 60192-3707 Tel.: +1 847 645-3600 Fax: +1 847 645-6201 servicebrc@boschrexroth-us.com repairbrc@boschrexroth-us.com	USA Central Region - Mitte Bosch Rexroth Corporation Electric Drives & Controls 1701 Harmon Road Auburn Hills, MI 48326 Tel.: +1 248 393-3330 Fax: +1 248 393-2906	USA Southeast Region - Südost Bosch Rexroth Corporation Electric Drives & Controls 2810 Premiere Parkway, Suite 500 Duluth, GA 30097 Tel.: +1 678 957-4050 Fax: +1 678 417-6637	**USA SERVICE-HOTLINE** 7 days x 24hrs - **+1-800-REXROTH** **+1 800 739-7684**
USA Northeast Region – Nordost Bosch Rexroth Corporation Electric Drives & Controls 99 Rainbow Road East Granby, CT 06026 Tel.: +1 860 844-8377 Fax: +1 860 844-8595	USA West Region – West Bosch Rexroth Corporation Electric Drives & Controls 7901 Stoneridge Drive, Suite 220 Pleasanton, CA 94588 Tel.: +1 925 227-1084 Fax: +1 925 227-1081	Canada East - Kanada Ost Bosch Rexroth Canada Corporation 5345 Outer Drive unit 5 Windsor, Ontario Canada N9A 6J3 Tel.: +1 519 737 7393 Fax.: +1 519 737 9999	Canada East - Kanada Ost Bosch Rexroth Canada Corporation Automation Division 3426 Mainway Drive Burlington, Ontario Canada L7M 1A8 Tel.: +1 905 335 5511 Fax: +1 905 335 4184 (Main) +1 905 335 9803 (Serv.) automation.service@boschrexroth.ca automation.repair@boschrexroth.ca
Canada West - Kanada West Bosch Rexroth Canada Corporation 5345 Goring St. Burnaby, British Columbia Canada V7J 1R1 Tel. +1 604 205 5777 Fax +1 604 205 6944 automation.service@boschrexroth.ca automation.repair@boschrexroth.ca	**Canada SERVICE-HOTLINE** 7 days x 24hrs - **+1 905 335 5511**	Mexico Bosch Rexroth Mexico S.A. de C.V. Calle Neptuno 72 Unidad Ind. Vallejo 07700 Mexico, D.F. Tel.: +52 55 57 54 17 11 Fax: +52 55 57 54 50 73 mario.francioli@boschrexroth.com.mx	Mexico Bosch Rexroth S.A. de C.V. Calle Argentina No 3913 Fracc. las Torres 64930 Monterrey, N.L. Tel.: +52 81 83 49 80 91 +52 81 83 49 80 92 +52 81 83 49 80 93 Fax: +52 81 83 65 52 80

Südamerika – South America

Argentina - Argentinien Bosch Rexroth S.A.I.C. "The Drive & Control Company" Rosario 2302 B1606DLD Carapachay Provincia de Buenos Aires Tel.: +54 11 4756 01 40 +54 11 4756 02 40 +54 11 4756 03 40 +54 11 4756 04 40 Fax: +54 11 4756 01 36 +54 11 4721 91 53 victor.jabif@boschrexroth.com.ar	Argentina - Argentinien NAKASE SRL Servicio Tecnico CNC Calle 49, No. 5764/66 B1653AOX Villa Balester Provincia de Buenos Aires Tel.: +54 11 4768 36 43 Fax: +54 11 4768 24 13 **Hotline: +54 11 155 307 6781** nakase@usa.net nakase@nakase.com gerencia@nakase.com (Service)	Brazil - Brasilien Bosch Rexroth Ltda. Av. Tégula, 888 Ponte Alta, Atibaia SP CEP 12942-440 Tel.: +55 11 4414 -56 92 +55 11 4414 -56 84 Fax sales: +55 11 4414 -57 07 Fax serv.: +55 11 4414 -56 86 alexandre.wittwer@rexroth.com.br	Brazil - Brasilien Bosch Rexroth Ltda. R. Dr.Humberto Pinheiro Vieira, 100 Distrito Industrial [Caixa Postal 1273] 89220-390 Joinville - SC Tel./Fax: +55 47 473 58 33 Mobil: +55 47 9974 6645 sergio.prochnow@boschrexroth.com.br
Columbia - Kolumbien Reflutec de Colombia Ltda. Calle 37 No. 22-31 Santafé de Bogotá, D.C. Colombia Tel.: +57 1 208 65 55 Fax: +57 1 269 97 37reflutec@etb.net.co			

Rexroth
Bosch Group

Bosch Rexroth AG
Electric Drives and Controls
P.O. Box 13 57
97803 Lohr, Germany
Bgm.-Dr.-Nebel-Str. 2
97816 Lohr, Germany
Phone +49 (0)93 52-40-50 60
Fax +49 (0)93 52-40-49 41
service.svc@boschrexroth.de
www.boschrexroth.com

R911309162

Printed in Germany
DOK-VISMOT-VM*-11VRS**-FK01-EN-P