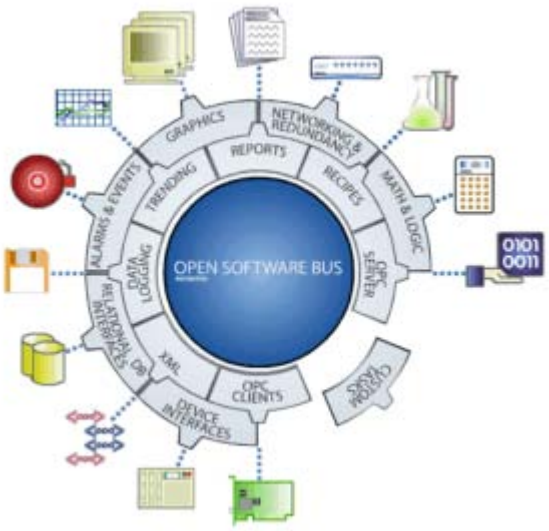
SIEMENS



FactoryLink Valmet Damaticx CIS Protocol Driver

for

FactoryLink 7.5.x

VALMET

Version 2.1
Printed: Friday, June 11, 2010

Valmet protocol driver

The VALMET protocol driver is a FactoryLink task, which supports communication with the Valmet Damatic computer and it's CIS interface. The driver is part of a FactoryLink system, which can be a single system or a redundant system. The driver communicates with one or more Valmet Damatic systems using a serial port for every device, or using a ethernet to serial switch. The protocol used is the Valmet Damatic Classic protocol.

FactoryLink is a registrated trademark of Siemens, Richardson Texas USA.

RLD Automation
Van Sonsbeeckstraat 11
5344 JB Oss
The Netherlands

tel. +31(0) 412 655 990
fax +31(0) 412 655 991
e-mail : support@rldautomation.eu

Version

Date	Version	State and changes
1994	1.0	Build for FL660
June-2009	2.0	Conversion driver to FL751, use of Ethernet serial switch added.
June-2010	2.1	Conversion driver to FL752, allow signal id to be set with command line parameter.

Table of Contents

This page is left blank intentionally.

1 Introduction

1.1 VALMET Protocol driver

The VALMET protocol driver is a FactoryLink task, which supports communication with the Valmet Damatic computer and it's CIS interface. The driver is part of a FactoryLink system, which can be a single system or a redundant system. The driver communicates with one or more Valmet Damatic systems using a serial port for every device, or using a ethernet to serial switch. The protocol used is the Valmet Damatic Classic protocol.

This manual is intended for use by a technician who is familiar with the FactoryLink software. Covered in this manual are the principles of operation, installation and configuration of the software. Included in the shipment is a demo-application, which can be used as an example of how to implement the features in your FactoryLink application.

1.2 Scope of this document

Please check the package you received with the checklist below. Should there be an item missing contact RLD Automation to correct the problem. There is a limit of 90 days after shipment to report problems!

This package includes the following:

- A VALMET protocol driver build contained on a CD-ROM or diskette(s).
- This manual.

You should also have:

- A MCI based translator e.g. the IO-Translator.
- The correct hardware and software for the serial port, or Ethernet to serial switdh on your platform.
- Honeywell, Siemens, Pilz, Symadyn or Texas Instruments PLC with ethernet communication board and software to program both the CPU and the CP boards.
- A serial connection between a Valmet Damatic system and the FactoryLink system, or an ethernet-network with at least both a Valmet Damatic system and the FactoryLink system connected to it.

1.3 References

[CIS]	DAMATIC Automation System, Process Control Computer Connection
[FL7_CONFIG]	FactoryLink 7.x.y Task Configuration Reference Guide
[PNI_ESPORT]	FPassport Networks, Ethernet Serial Server Users Manual

This page is left blank intentionally.

2 Driver principle

2.1 The IO-translator RAP Driver principle

RAPD stands for Rapid Application Protocol Driver. The RAPD principle was adopted so that protocol drivers can be easily and rapidly configured for a FactoryLink application. RAPD is based on the Mailbox Communication Interface standard or MCI, which defines how a protocol driver task communicates with an I/O Translator task. The RAPD system consists of a protocol driver that communicates with external devices (RTUS, PLC's, etc.) and a translator that controls data storage (going to and coming from a protocol driver) in the FactoryLink real-time database. All data collected by the protocol driver is referenced as contiguous blocks or ranges within the device. This enables communication between the driver and a device to be very efficient. All data is referenced between the driver and the translator in terms of datasets. Datasets, described in the next section, define memory regions or locations of data within a device.
Please check the package you received with the checklist below. Should there be an item missing contact RLD Automation bv to correct the problem. There is a limit of 90 days after shipment to report problems!

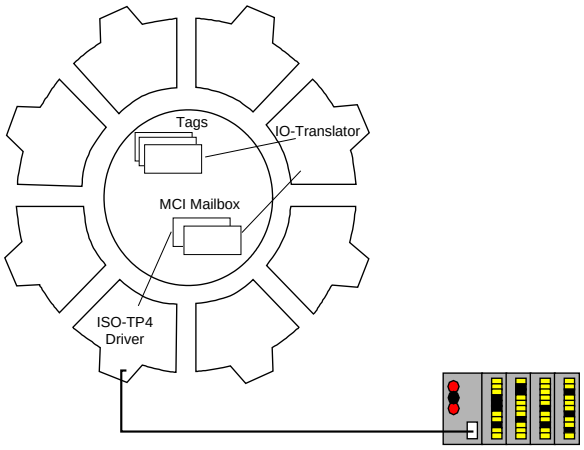


Figure 2.1 The RAPD principle.

The protocol driver and the I/O Translator communicate with one another through FactoryLink mailbox tags, according to the MCI standard. Every task (translator and protocol driver) has its own mailbox, so for full communication between a translator and a protocol driver a mailbox database element for every task has to be defined. The MCI standard is especially designed for the following situation. To use one translator and several protocol drivers. For example the translator together with the ISO-TP4 driver and the Modbus Plus protocol driver. Aside from storage duties, the translator provides data conversions (i.e. analog, IEEE conversions, etc.) for I/O data to/from a protocol driver.

2.1.1 FactoryLink domain selection

The standard domain for the VALMET driver is the SHARED domain. The protocol driver communicates with a dedicated piece of hardware, therefore only one task should be able to access the hardware. If only one

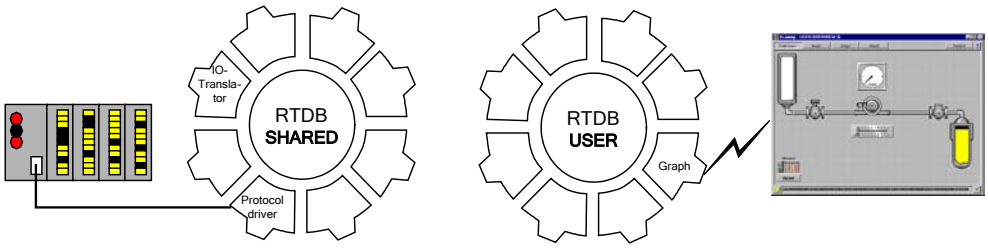


Figure 2.2 Domain selection for the driver.

program accesses the hardware, the task should be located in the shared domain and therefore started by the shared runtime-manager.

Important: The protocol driver and the translator must be in the same domain (either SHARED or USER).

2.2 FactoryLink task definition

The protocol driver must, with the FactoryLink Configuration Explorer (FLCE) or the FactoryLink Configuration Manager (FLCM), be entered in the System Configuration table. An entry of an existing task which will not be used at run-time can be overwritten or a new entry can be created with (as a minimum) the following data:

Task Name	*Description ....*	*Executable File*
VALMET	***VALMET Protocol Driver***	***bin/valmet***

The Task Name and name of the executable file must be filled in as shown above, the FactoryLink kernel uses these entries to start the driver, and to confirm that the driver registers correctly to the FactoryLink kernel.

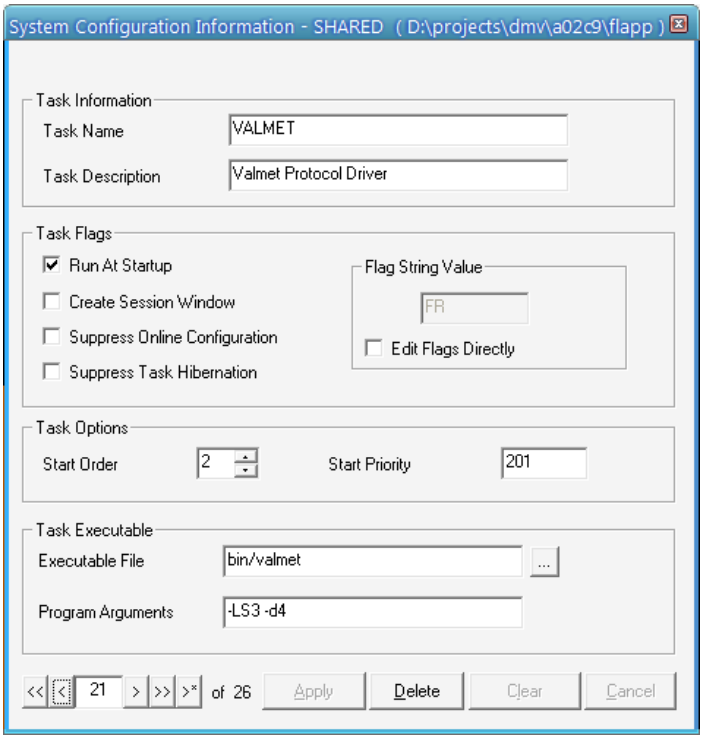


Figure 2.3 Task definition of the VALMET driver in the configuration explorer.

Important: The protocol driver needs two entries in the system configuration panel, one defined as in the figure above, and one 'free' (or empty) task definition. If the driver can not find a free task definition when starting, the driver task will exit with an error. Practically this means: there must be less then 31 task configured in the FactoryLink system configuration table.

2.3 Devices

Every Valmet Damatic Computer connected to the driver via network or serial cable is called a device, this term is defined in the MCI standard as a logical external device. A Valmet Damatic Computer is called a device in the FactoryLink application and has a logical name (only used in the application configuration). A serial connection is defined by the COM port used by the device, using a serial connection over TCP/IP the device is identified by the IP adres of the Ethernet-Serial switch.

2.4 Communication functions

Data exchange between a FactoryLink workstation and a Valmet Damatic Computer is performed by using the *read*- and *write*-services of the CIS interface. The protocol driver task uses the *read*-service to read (or fetch) data from the Valmet Damatic Computer, and it uses the *write*-service to write data to the Valmet Damatic Computer. Both actions are always initiated by the driver task.
The CIS interface supports different types of read and write services, the ones supported are described in the next paragraphs. In the table below are all the possible services listed together with the information if they are supported or not.

Protocol	Read/Write Service	Supported
CIS interface	Block read	Yes
	Block write	No
	Exception write	No
	Encoded write	No
	Unsolicited Receive	No

2.5 Datasets

Data exchange is, according to the definitions stated in the MCI standard, is based upon datasets. A dataset is a contiguous area, such as a register or memory location, of data in the external device. All datasets are defined as digital tags. The protocol driver defines all specifics, such as data type about datasets, starting address, or length, while the IOX task references the dataset by tag name for read/write operations. When the IOX task initiates a trigger on a dataset, the MCI system notifies the protocol driver that an event has occurred on that dataset. The event notification, in turn, causes a protocol specific message to be generated and sent to an external device. The scenario works in reverse when the protocol driver receives unsolicited data from a device.

Dataset	Maps to external device
Referenced by dataset control tag	Memory location, defined by area type, start address, and length
Tag 1	Address: n
Tag 2	Address: n + 1
Tag 3	Address: n + 2
Tag 4	Address: n + 3
Tag x	Address: n + m

A dataset can be read or written to with one command or trigger. With the exception of actions on the complete dataset, most protocol drivers are capable of addressing specific elements or a group of elements in the dataset. How these actions are performed depends on the size of a basic element in the dataset, which can be bit, byte, word, or long, depending on the data area in the device or the communication protocol.

2.6 Block read

The block read commands are performed on a contiguous block of data in the external device, specified by datasets. The FactoryLink station initiates these commands. The FactoryLink application developer configures digital triggers for the IO-translator task to start the read commands. Data received from a device as a result of a read request is stored in FactoryLink tags.

2.7 Multithreading of the driver

The VALMET driver is a high performance RAP driver, for speed considerations the driver is build as a multi-threaded task. On start-up there is initially one thread active, this thread is the ‘main’ thread, first it will load the configuration defined with the FactoryLink Configuration Explorer. In the configuration are all the device definitions found, next the protocol driver starts a thread for every device or PLC. One thread for every device is needed to perform the ‘read’ commands, a second thread executes all the ‘write’ commands and the third thread waits for unsolicited receives. In combination with MCI the first two threads will wait for any read or write command to occur. If there is a command it is immediately executed and the response is send to the I/O translator.
An additional thread, the listen thread, is started by the driver, this thread waits for incoming connection requests from devices, this thread only accept connection requests from devices supporting the unsolicited receive. The driver initiates connection requests for the read and write functionality.

The total number of threads for the device will be: **1 * number of devices + 1**. The extra thread is the main thread, responsible for dispatching the read and write commands from the I/O translator to every read thread of a device.
The logical order of starting and creating threads is the following:

1. Task is started, one thread present. Configuration data is loaded, among it are the device definitions.
2. For every device a thread for the read command is started (MCI).

This page is left blank intentionally.

3 Installation

3.1 System requirements

The VALMET driver will install and run on any IBM PC or PC-compatible computer equipped with:

- A processor equivalent to an Intel 80486 or later. RLD Automation recommends at least an Intel Pentium-class or compatible processor.
- A CD-ROM.
- At least 5 MB of free hard disk space.
- Microsoft Windows XP/2003, with at least service pack 1 installed.
- SIEMENS FactoryLink 7.5.0.

3.2 VALMET driver installation

The main objective of the VALMET driver installation is copying a group of files from the installation media to your computer system. In this group all the files related to the FactoryLink environment are copied to your FactoryLink system directory (normally set by the 'FLINK' environment variable), including both the runtime and configuration options.
To start the installation procedure: locate the 'setup.exe' file on the installation media, and execute the setup utility.

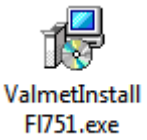


Figure 3.1. Setup the application (large icon view).

After starting the setup, some questions have to be answered, the most important one is where your FactoryLink system directory is located. The setup assumes that the environment variable %FLINK% is correct, but gives you the opportunity to change the directory to install to. Most steps of the installation process are presented in the following figures.

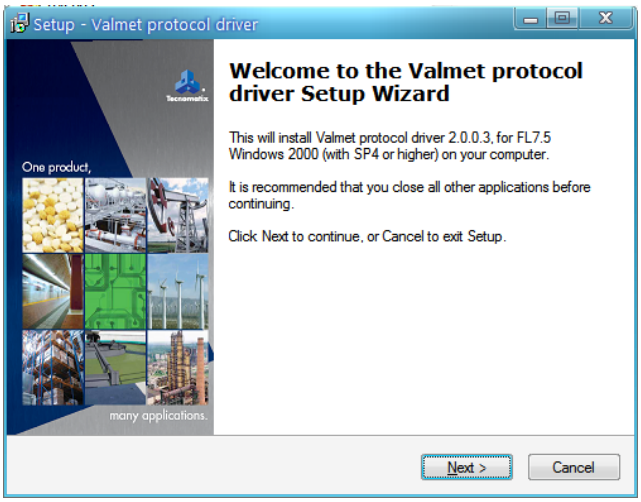


Figure 3.2. The welcome wizard panel.

The first screen displayed is the 'Welcome' screen, click 'Next >' to continue. The next panel of the setup wizard displays the VALMET driver end-user license agreement. Read this agreement carefully. If you install the VALMET driver you agree to abide by the terms of the license.
If you do not agree to the license terms, click 'I do not accept the agreement' folowed by 'Cancel'. Setup will quit immediately. Otherwise, click 'I accept the agreement' followed by 'Next >' to continue. The next screen will list the minimum requirements for your system needed by the VALMET driver. It also explains how to choose the program location in the next screen, press 'Next >' when you are ready.

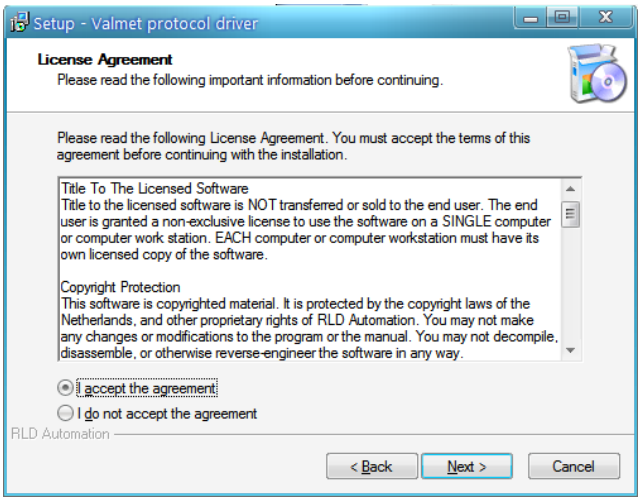


Figure 3.3. The software license agreement.

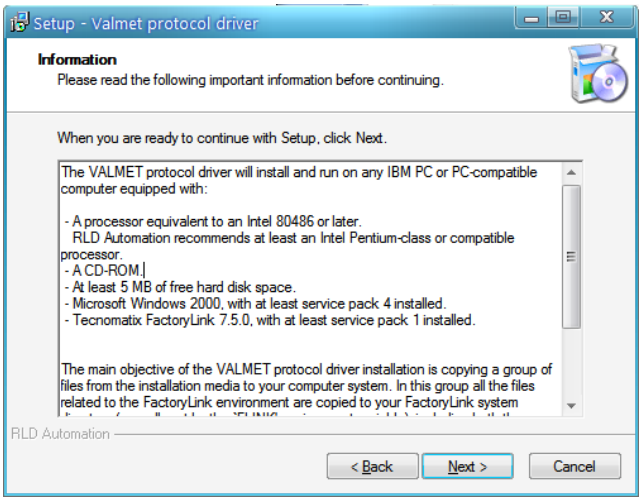


Figure 3.4. Additional informaton.

Use the browse window to specify a directory for the FactoryLink system files, in case the one selected by the set-up wizard is not the one where you want to install this FactoryLink driver.
Next select the components you wish to install, the VALMET driver has three components. First the driver itself this component can not be deselected, it is always installed. The optional components are the manual in pdf-format and support files for debugging purposes. The last option includes the program 'debug viewer' from sysinternals.

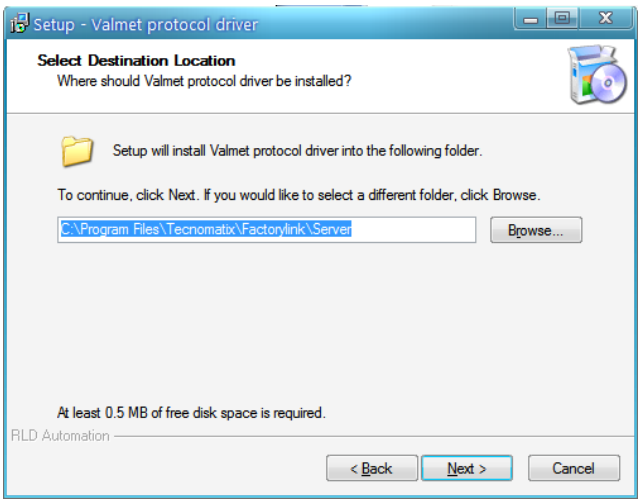


Figure 3.5. Select the destination directory.

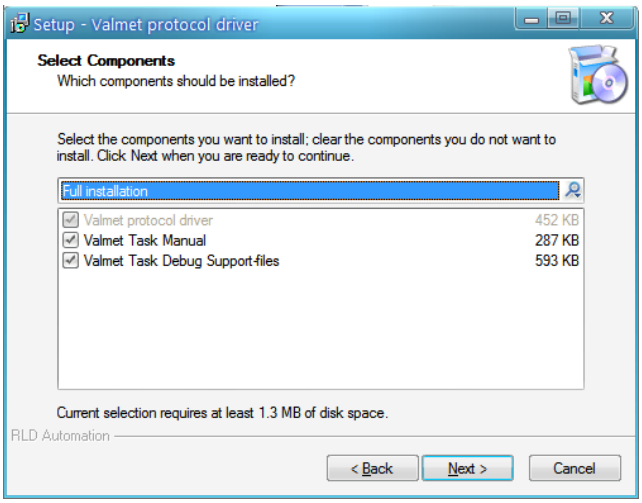


Figure 3.6. Select the components to install.

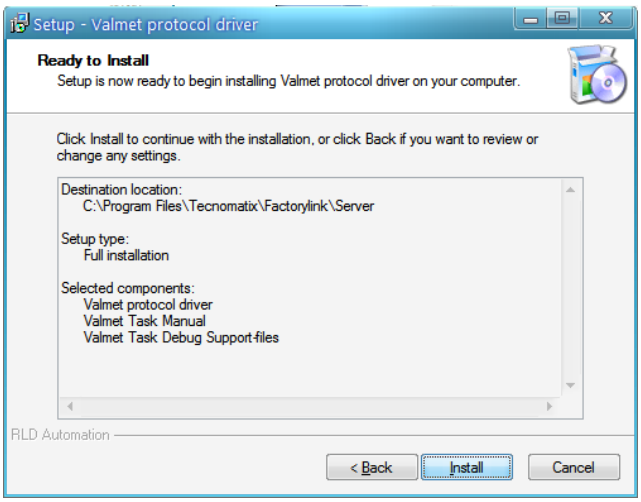


Figure 3.7. Ready to install the application.

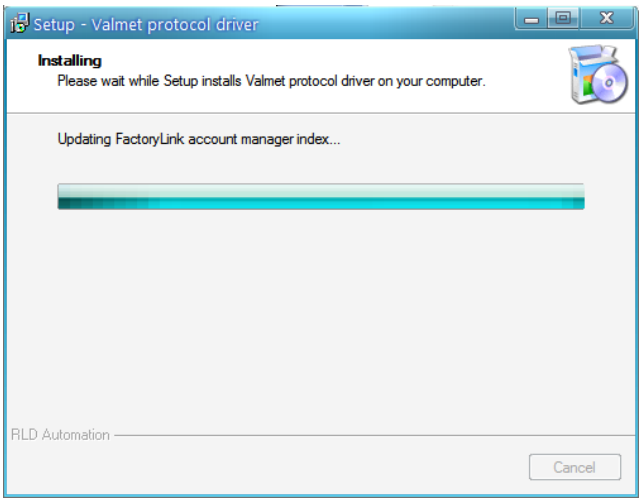


Figure 3.8. Installation progress.

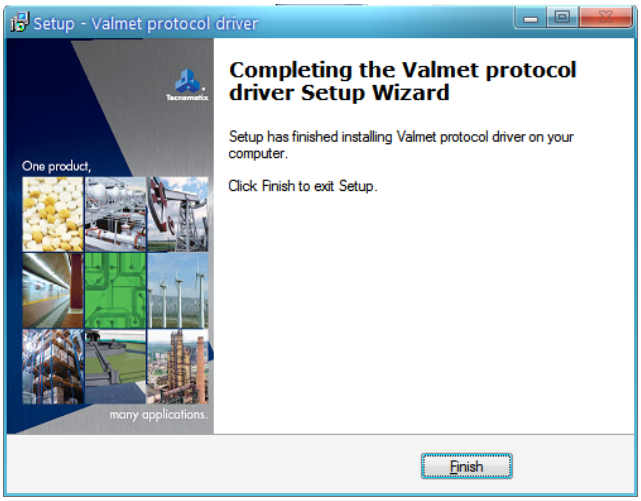


Figure 3.9. Installation is completed.

You need to confirm your settings and requirements, before the actual installation process is started. After completion of the installation process, the last page of the wizard is shown, and pressing the 'Finish' button closes the setup window.

3.3 Installation of the serial port

It is necessary to have a serial port on your system which can be used by the protocol driver for communicating with the Valmet Damatic CIS system(s). Refer to the manual(s) of your system for installing/setting up a serial port. Make sure everything is installed correctly, and additional device drivers, when needed, are loaded. Connect the serial port to the Valmet Damatic CIS system(s), connector layouts are described in the manuals for your system and the Valmet system. Refer to reference [CIS] for information about setting the serial communication parameters used by the Valmet Damatic driver.

This page is left blank intentionally.

4 VALMET driver configuration

4.1 FactoryLink task configuration

The configuration of the VALMET driver consists of three main parts, two of these parts need some configuration, the third part has already a default configuration. This default configuration will in most cases suit the needs of every application. The configuration main parts are: Run time manager configuration, VALMET driver and IO-translator table configuration, the third part is the INI-file of the driver.
The first (Run time manager configuration) and the second (driver table configuration) configuration, are normally done with the configuration explorer (FLCE), and involves changing or adding settings in configuration tables. Third part of the configuration is the task's INI-file, a default file is already supplied during installation, changes in this file influences your system, not only the application. The file is located in your FactoryLink system directory (%FLBIN%), and not part of an application.
Note: The VALMET driver does not need an ini-file, there is even not one installed on your FactoryLink system.

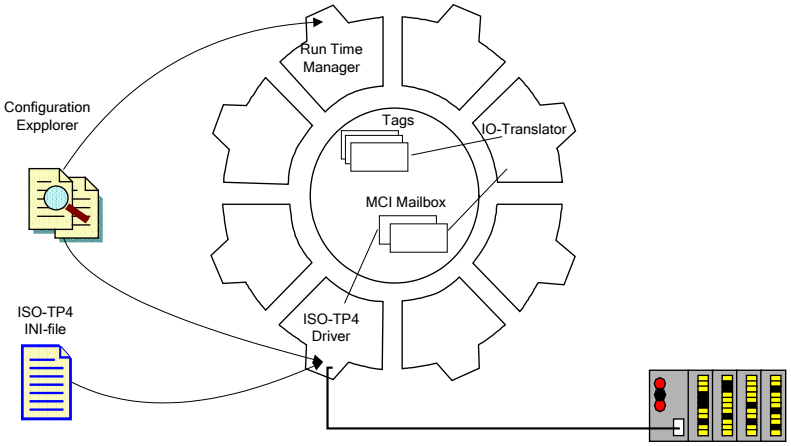
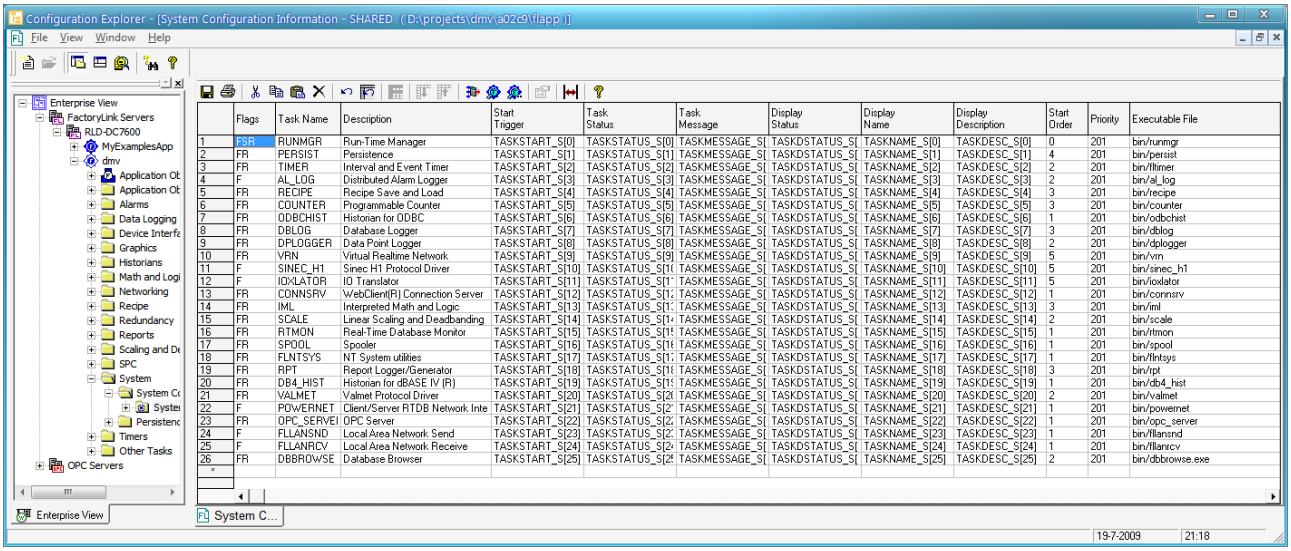


Figure 4.1 VALMET driver configuration.

4.2 Run time manager configuration

In order to activate the VALMET driver as a FactoryLink task it needs some configuration in the 'system configuration' panel of the FactoryLink Configuration Explorer.



	Flags	Task Name	Description	Start Trigger	Task Status	Task Message	Display Status	Display Name	Display Description	Start Order	Priority	Executable File
1	FSR	RUNMGR	Run-Time Manager	TASKSTART_S[0]	TASKSTATUS_S[0]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[0]	TASKDESC_S[0]	0	201	bin/runmgr
2	FR	PERSIST	Persistence	TASKSTART_S[1]	TASKSTATUS_S[1]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[1]	TASKDESC_S[1]	4	201	bin/persist
3	FR	TIMER	Interval and Event Timer	TASKSTART_S[2]	TASKSTATUS_S[2]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[2]	TASKDESC_S[2]	2	201	bin/fltimer
4	F	AL_LOG	Distributed Alarm Logger	TASKSTART_S[3]	TASKSTATUS_S[3]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[3]	TASKDESC_S[3]	2	201	bin/al_log
5	FR	RECIPE	Recipe Save and Load	TASKSTART_S[4]	TASKSTATUS_S[4]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[4]	TASKDESC_S[4]	3	201	bin/recipe
6	FR	COUNTER	Programmable Counter	TASKSTART_S[5]	TASKSTATUS_S[5]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[5]	TASKDESC_S[5]	3	201	bin/counter
7	FR	ODBCHIST	Historian for ODBC	TASKSTART_S[6]	TASKSTATUS_S[6]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[6]	TASKDESC_S[6]	1	201	bin/odbchist
8	FR	DBLOG	Database Logger	TASKSTART_S[7]	TASKSTATUS_S[7]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[7]	TASKDESC_S[7]	3	201	bin/dblog
9	FR	DPLOGGER	Data Point Logger	TASKSTART_S[8]	TASKSTATUS_S[8]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[8]	TASKDESC_S[8]	2	201	bin/dplogger
10	FR	VRN	Virtual Realtime Network	TASKSTART_S[9]	TASKSTATUS_S[9]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[9]	TASKDESC_S[9]	5	201	bin/vrn
11	F	SINEC_H1	Sinec H1 Protocol Driver	TASKSTART_S[10]	TASKSTATUS_S[10]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[10]	TASKDESC_S[10]	5	201	bin/sinec_h1
12	F	IOXLATOR	IO Translator	TASKSTART_S[11]	TASKSTATUS_S[11]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[11]	TASKDESC_S[11]	5	201	bin/ioxlator
13	FR	CONNSRV	WebClient(R) Connection Server	TASKSTART_S[12]	TASKSTATUS_S[12]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[12]	TASKDESC_S[12]	1	201	bin/connsrv
14	FR	IML	Interpreted Math and Logic	TASKSTART_S[13]	TASKSTATUS_S[13]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[13]	TASKDESC_S[13]	3	201	bin/iml
15	FR	SCALE	Linear Scaling and Deadbanding	TASKSTART_S[14]	TASKSTATUS_S[14]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[14]	TASKDESC_S[14]	2	201	bin/scale
16	FR	RTMON	Real-Time Database Monitor	TASKSTART_S[15]	TASKSTATUS_S[15]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[15]	TASKDESC_S[15]	1	201	bin/rtmon
17	FR	SPOOL	Spooler	TASKSTART_S[16]	TASKSTATUS_S[16]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[16]	TASKDESC_S[16]	1	201	bin/spool
18	FR	FLNTSYS	NT System utilities	TASKSTART_S[17]	TASKSTATUS_S[17]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[17]	TASKDESC_S[17]	1	201	bin/flntsys
19	FR	RPT	Report Logger/Generator	TASKSTART_S[18]	TASKSTATUS_S[18]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[18]	TASKDESC_S[18]	3	201	bin/rpt
20	FR	DB4_HIST	Historian for dBASE IV (R)	TASKSTART_S[19]	TASKSTATUS_S[19]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[19]	TASKDESC_S[19]	1	201	bin/db4_hist
21	FR	VALMET	Valmet Protocol Driver	TASKSTART_S[20]	TASKSTATUS_S[20]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[20]	TASKDESC_S[20]	2	201	bin/valmet
22	F	POWERNET	Client/Server RTDB Network Inte	TASKSTART_S[21]	TASKSTATUS_S[21]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[21]	TASKDESC_S[21]	1	201	bin/powernet
23	FR	OPC_SERVEI	OPC Server	TASKSTART_S[22]	TASKSTATUS_S[22]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[22]	TASKDESC_S[22]	1	201	bin/opc_server
24	F	FLLANSND	Local Area Network Send	TASKSTART_S[23]	TASKSTATUS_S[23]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[23]	TASKDESC_S[23]	1	201	bin/fllansnd
25	F	FLLANRCV	Local Area Network Receive	TASKSTART_S[24]	TASKSTATUS_S[24]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[24]	TASKDESC_S[24]	1	201	bin/fllanrcv
26	FR	DBBROWSE	Database Browser	TASKSTART_S[25]	TASKSTATUS_S[25]	TASKMESSAGE_S[	TASKDSTATUS_S[	TASKNAME_S[25]	TASKDESC_S[25]	2	201	bin/dbbrowse.exe

Figure 4.2 FactoryLink task configuration.

The required settings for the VALMET driver are its task name and executable name, the last one is not visible in the figure above, but can be reached by panning through the configuration row. The following settings apply:

Task Name = VALMET
Executable File = bin/valmet

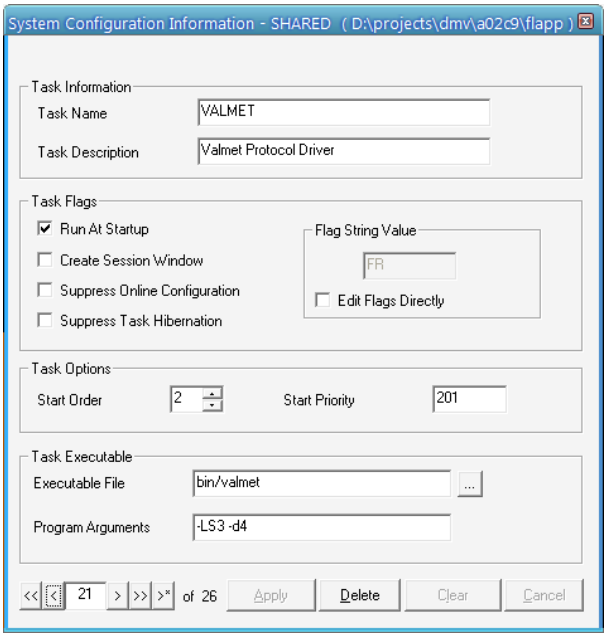


Figure 4.3 VALMET task definition in the configuration explorer.

Important: The protocol driver needs two entries in the system configuration panel, one defined as in the figure above, and one 'free' (or empty) task definition. If the driver can not find a free task definition when starting, the driver task will exit with an error. Practically this means: there must be less then 31 tasks configured in the FactoryLink system configuration table.

4.2.1 Program arguments

-LS*n* Local Station id with number *n*, needed if the protocol driver and IO-translator reside on different nodes (or FactoryLink workstations). This must be an unique (station) number for the network.
Two MCI tasks (the IOX task and a protocol driver) can run on different computers communicating with one another. Because the two computers can differ in data direction (different platforms), MCI supplies information specifying where the dataset came from and what kind of data orientation it has.
The station identification is a unique number of the host station on the LAN. The user must assign this number and keep it unique over the LAN. The station identification can be passed to the driver task as a program argument.
If tasks are to support communications over a LAN, some information must be exchanged between the tasks before they can begin communications. The tasks are probably running on separate nodes in two different FactoryLink applications, which means all tag ids are most likely not the same on both systems. MCI uses the dataset control tag for identification of the dataset, exchanging the tag-ids between two systems does not always result in identification of the same tag!. The –LS option tells the driver to use the tag-name instead of the id to identify the dataset control tag.
The –LS option accepts environment variables for the *n* argument, the name of the environment variable must be enclosed in curly brackets, eg {FLSTATION}, where the environment variable FLSTATION is defined as a number.

-d*n* Debug option, the level of debug information is set with the number n. The range of this number is from 1 until 9. If no number is specified the default level will be 1. The debug output will be visible in the 'window' of the protocol driver, that of the run time manager.

-l*n* Same as the previous option, difference is the output device. For this option an ASCII log file is generated, being the file: {FLAPP}/{FLNAME}/{FLDOMAIN}/log/valmet.log

-v*n* Same as the previous option, difference is the output device. For this option the output is send to the default debug-viewer on the system. If no debug-viewer is present or active on the system, the 'SysInternals' program 'dbgview.exe' can be used to monitor the debug output (www.sysinternals.com). This program can be installed as an option to the protocol driver.

-m*n* Internal buffersize for every thread, the buffersize *n is in Kbytes.*

-signal*n* Internal resource, used for accessing the RTDB elements, default the signal id used for intertask flagging is'25'. This setting can be overwritten with this argument, the value can range form 20 – 31; specifying a value outside this range will reinforce the default value 25.

4.3 INI-file

The VALMET driver does NOT use an INI-file.

4.4 FactoryLink domain selection

The standard domain for the Valmet Damatic protocol driver is the SHARED domain. The protocol driver communicates with a dedicated piece of hardware, there fore only one task should be able to access the hardware. If only one program accesses the hardware, the task should be located in the shared domain and therefore started by the shared runtime-manager.

Important: The protocol driver and the translator must be in the same domain (either SHARED or USER).

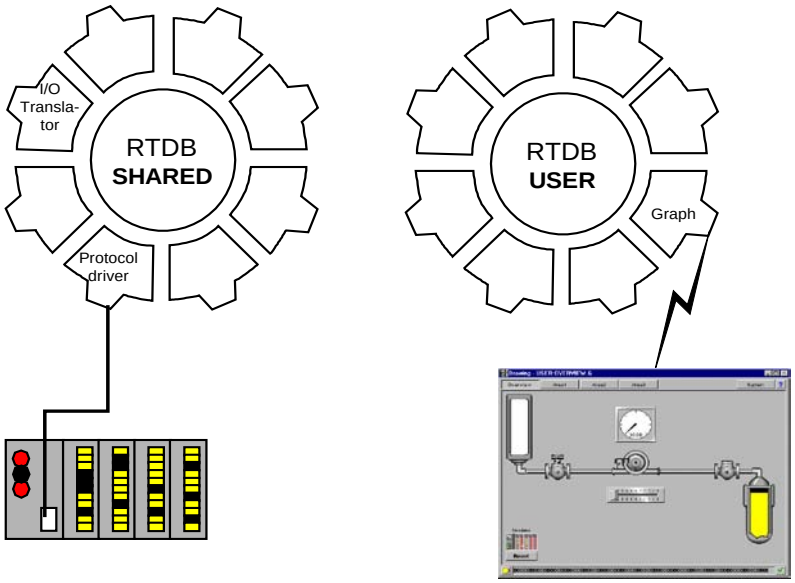


Figure 4.4 *Standard domain selection.*

4.5 Valmet Damatic Classic Tag Address Format

Internally, the Valmet GCU addresses its points by a combination of Bus Number (1-15), Process Station's Number (1-14), Block (0-7), Signal Group (0-3) and Signal Number (0-3).

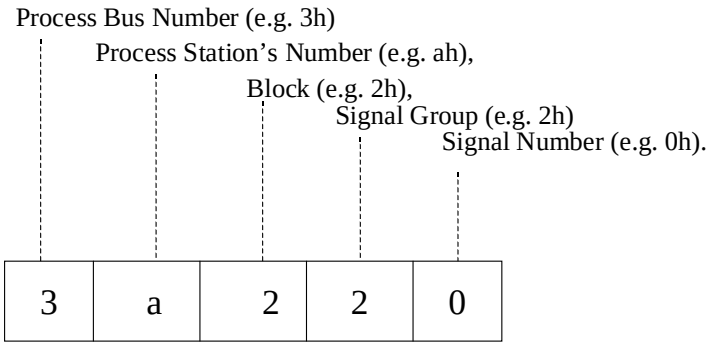


Figure 4.5 *Classic tag address format.*

4.6 Configuration tables

The configuration tables for the VALMET driver can be accessed with the FactoryLink Configuration explorer. For general information about entering data in FactoryLink configuration tables refer to the FactoryLink Manual. In the Configuration Explorer select ‘Device Interfaces’.

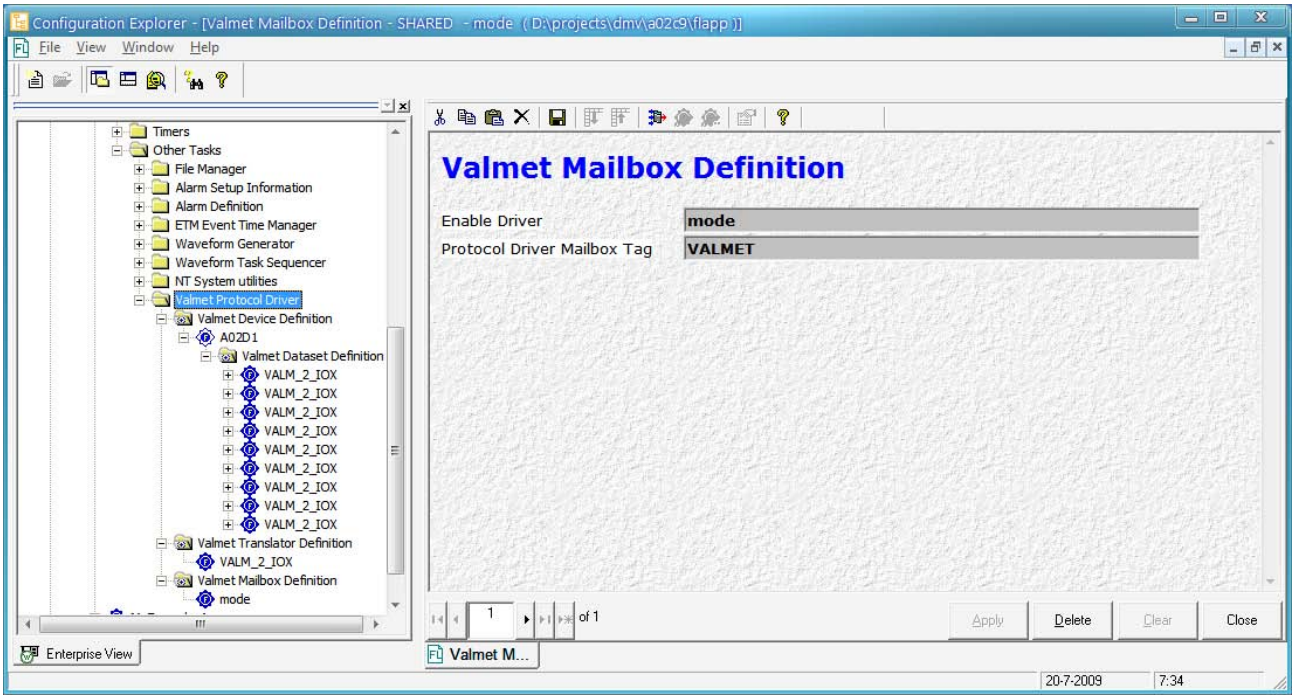


Figure 4.6. FactoryLink Configuration explorer.

A table entry appears: ‘VALMET Protocol Driver. This entry gives access to the configuration tables, three tables are present, ‘VALMET Device definition’, ‘VALMET Translator Definition’ and ‘VALMET Mailbox D’. The third table needs only one row to configure the mailbox used by the protocol driver. The second table defines all the translator used by the protocol driver, the configuration defaults to one row: the mailbox tag used by the IO-translator. The first table configures all the devices (PLC’s) that will be present in the network, a sub table defines the datasets for every (selected) device.

Note: For general information about entering data in FactoryLink configuration tables, refer to the FactoryLink Fundamentals Manual.

4.6.1 Valmet Mailbox definition

The ‘VALMET Driver Definition’ panel is used to define global settings for the driver. For these global settings only one record or row is needed, if you define more then one row, the driver will ignore the second and following rows. The first row however is required, an empty table will generate a task error and cause the driver to stop. The field ‘Protocol Driver Mailbox Tag’ is required, without a mailbox the drive is unable to function.

Figure 4.7. Configuration panel VALMET mailbox definition.

Enable Driver

Tag name of a digital tag, the value of this tag determines if the driver is enable or disabled. A disabled driver does not communicatie with the Valmet device. This field is optional and left empty, the driver is enabled and will function normal. A tag value ON enables the driver, the driver is disabled with a taga value OFF.

entry Optional
entry type Tag name, tag type: DIGITAL

Protocol Driver Mailbox Tag

Tag name of the Valmet protocol mailbox element that the developer defines, to be referenced by a IO translator task (**N.B.** MCI must be supported by the IO translator task). The IO translator task uses this mailbox to send requests to the Valmet protocol driver.

entry ✕ Required
entry type Tag name, tag type: MAILBOX

4.6.2 Valmet translator definition

From the display of all the panels, select the *Valmet Translator Definition* panel.

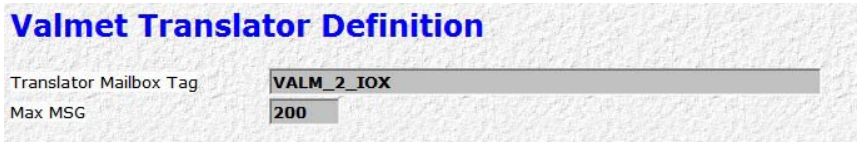


Figure 4.8. Configuration panel VALMET translator definition.

The Valmet translator definition panel allows the user to specify one or more decoder(s). Every Mailbox Tag specifies a different decoder. Only one Mailbox Tag, for every decoder, is needed for full performance of MCI. Specify the following information.

Translator Mailbox Tag

Tag name of a translator mailbox element, to be referenced by the VALMET protocol driver (**N.B.** MCI must be supported by the translator task). The translator task uses this mailbox to receive data from the VALMET protocol driver.

entry ✕ Required
entry type Tag name, tag type: MAILBOX

Max MSG

The maximum number of requests for the protocol driver, which can be queued in the mailbox tag. Recommended is a value of 100 messages. The number of messages is limited by the size of an integer value (9999), and practically by the amount of available memory. The memory needed for a request depends on the size of the configured datasets.

entry ✕ Required
entry type Numerical value, 1..9999
default If not configured the value defaults to 100 messages

4.6.3 Valmet Device definition

The Valmet Device definition panel allows the user to specify logical devices for the communication. Each logical device uses exclusively a serial communication port. Laos, it is possible to specify tags that enable the programmer to enable/disable reading and writing to the physical device. Specify the following information.

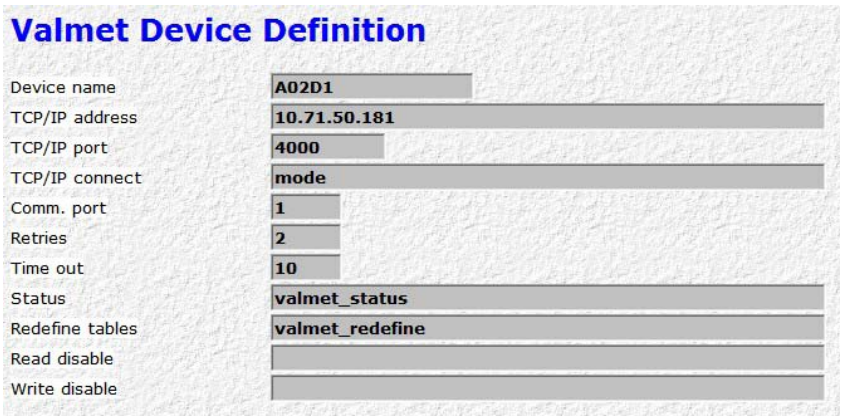


Figure 4.9 Configuration panel ISO-TP4 device definition.

Device Name

Logical name assigned by the user to represent a particular communication device. This field is used as a selection criterion for the next panel: ISO-TP4 Dataset Definition.

entry ✘ Required
entry type Alphanumeric string of up to 16 characters

TCP/IP address

The Valmet Damatic Computer supports only a serial protocol on the CIS interface, the driver supports the use of an Ethernet-serial switch to connect the serial port through an Ethernet connection. The TCP/IP address of the Ethernet-Serial switch, connecting the FactoryLink station and the Valmet Damatic station, is specified in this field. The field is only used when connecting the Valmet Damatic Computer through Ethernet. The Ethernet address is assumed to be in the format: “192.168.2.1”.

entry Optional
entry type Alphanumeric string of up to 48 characters

TCP/IP port

The Valmet Damatic Computer supports only a serial protocol on the CIS interface, the driver supports the use of an Ethernet-serial switch to connect the serial port through an Ethernet connection. The TCP/IP port used by the Ethernet-Serial switch, connecting the FactoryLink station and the Valmet Damatic station, is specified in this field. The field is only used when connecting the Valmet Damatic Computer through Ethernet.

entry Optional
entry type Numerical value

TCP/IP connect

Tag name of a digital tag, the value of this tag determines if the device is connected to or disconnect from the Ethernet-serial switch. Disabling the connection will stop the communication between the driver and Valmet device. This field is optional and left empty, the TCP/IP connection is enabled. A tag value ON enables the driver to connect to the Ethernet-serial switch, the connection is disabled with a tag value OFF. The field is only used when connecting the Valmet Damatic Computer through Ethernet.

entry Optional
entry type Tag name, tag type: DIGITAL

Comm. port
The number of the serial port through which the device must be accessed. Field is required in case the Ethernet address of the Ethernet-serial switch is left empty.

entry Optional
entry type Alphanumeric string, 1..16.

Retries
The number of retries that the protocol driver will execute when the Valmet computer does not respond or responds negatively to a request.

entry Required
entry type Numerial value.

Timeout
The time out value for the protocol driver, specified in seconds. The protocol driver will wait for the full response to be received from Valmet during the specified number of seconds. If there is no (valid) response within the time out period the driver will retry.

entry Required
entry type Numerial value.

Status
Real-time database element written to by the Valmet protocol driver task to indicate the status of a communication command. The status element can be referenced by any task to handle communication error situations. For a detailed list of communication errors see chapter 5.

entry Optional
entry type Tag name, tag type: ANALOG

Redefine tables
Real-time database element forced to ON by the user to trigger a rebuild of the set definitions in the Valmet computer. The rebuild is performed in case the database element has the value ON and the change flags are ON. Used in case the Valmet computer has powered on or has lost the definitions downloaded by the driver through some other cause.

entry Optional
entry type Tag name, tag type: DIGITAL

Read disable
Real-time digital database element used to enable/disable read commands for the logical device. Read commands are enabled in case there is no tag defined, or the status of the digital tag is OFF. Read commands are disabled if the status of the tag is ON.

entry Optional
entry type Tag name, tag type: DIGITAL

Write disable
Real-time digital database element used to enable/disable write commands for the logical device. Write commands are enabled in case there is no tag defined, or the status of the digital tag is OFF. Fetch commands are disabled if the status of the tag is ON.

entry Optional
entry type Tag name, tag type: DIGITAL

4.6.4 Valmet Dataset definition

The Valmet Dataset definition panel allows the user to specify datasets for a particular logical device. Different logical device names are selected in the panel *Valmet Device definition*. Specify the following information.

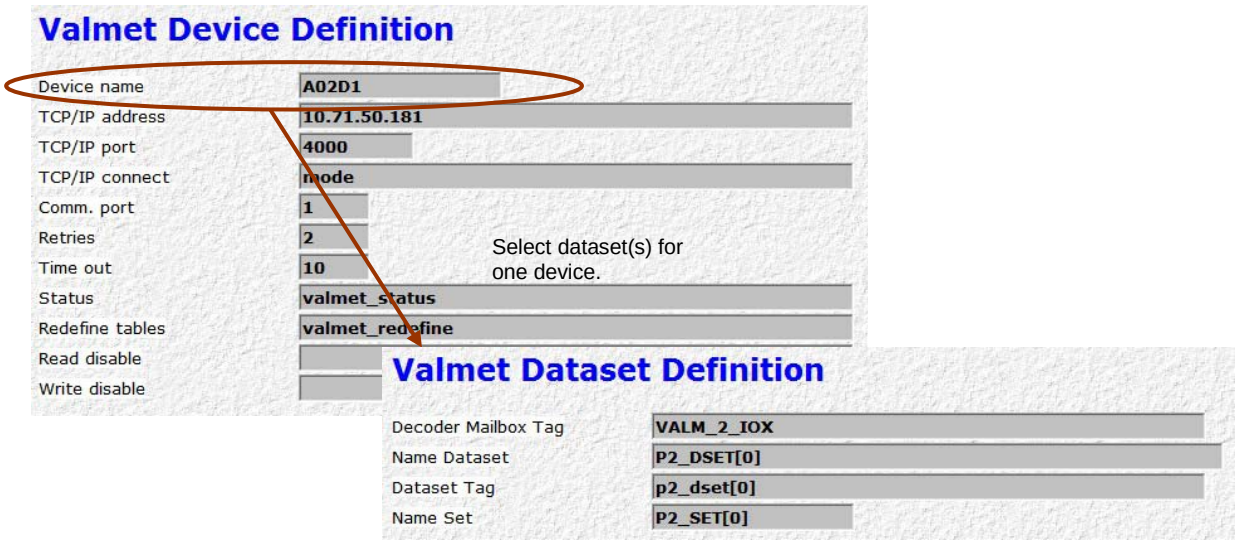


Figure 4.10 Configuration panel VALMET dataset definition.

A dataset is a contiguous data area in the device, and is identified with its type, start and length. The IO-Translator mailbox defines the IO-Translator to use for this data set. The maximum number of messages in this mailbox at any given time before the driver stops writing messages into the mailbox is defined with the IO-Translator mailbox.
The dataset tag is a digital tag used to identify the dataset, the same data set tag will be present in the configuration for the IO-Translator. The user is responsible for defining an unique tag name for every data set.

IO-Translator Mailbox Tag

Tag name of a translator mailbox element, to be referenced by the VALMET protocol driver task (**N.B.** MCI must be supported by the translator task). The I/O translator task uses this mailbox to receive data from the VALMET protocol driver.

entry ✗ Required
entry type Tag name, tag type: MAILBOX

Name Dataset

Unique name for the data set. The name is used internally in the decoder and protocol driver for activating a data fetch from the external device. A dataset is referenced, internally by the decoder and the protocol driver, by the name. Therefore the name specified in this field must be unique.

entry ✗ Required
entry type Alphanumeric string of up to 18 characters

Dataset Tag

Tag name representing a (unique) logical name for a dataset. The tag is used internally in the I/O translator and protocol driver for activating a dataset read from the external device. The I/O translator and the protocol driver internally reference a dataset by the tag name defined in this field. As a consequence the tag specified must be unique. The same tag should be used in configuration of the I/o translator to select the conversions for the dataset.

entry ✗ Required
entry type Tag name, tag type: DIGITAL

Name Set

Unique name for the set that defines the actual contents of this data set. This name is used to link to the set definition panels. The name is used internally in the protocol driver. Therefore the name specified in this field must be unique.

entry ✕ Required/Optional
entry type Alphanumeric string of up to 18 characters

The name set is optional, actually one does not need to define a name set, if the Dataset name is 'DEVX'. The dataset name 'DEVX' is reserved to retrieve internal data from the driver. In the following table is the dataset defined.

Element dataset 'DEVX'	Description	Type
Device[20]	Device name.	String (20)
TCPIP[20]	TCP/IP address, empty for direct serial connection.	String(20)
TCPport	TCP/IP port number, zero for direct serial connection.	Word
TCPconnection	Indication if TCP connection is active (1)or disconnected (0), always zero for direct serial connection.	Word
COMport	Number of the COMM port for a direct serial connection.	Word
Status	Status of the driver	Word
Driverenable	Indication of driver enabled (1) or disabled (0)	Word
TCPenable	Indication of TCP connection enabled (1) or disabled (0)	Word
Disable[2]	Two integers representing the read and write disable of the device; index 0 is read disable and index 1 is the write disable.	Word
Count[2]	Two counters representing the number of reads performed (index 0) ands the number of writes performed (index 1)	Long

The dataset mentioned above is the only dataset that can be written. To write to the dataset 'DEVX', no values are needed. Writing to the dataset resets the read and write counters.

4.7 Valmet Nameset definition

From the display of all the panels, select the *Valmet Nameset definition* panel.

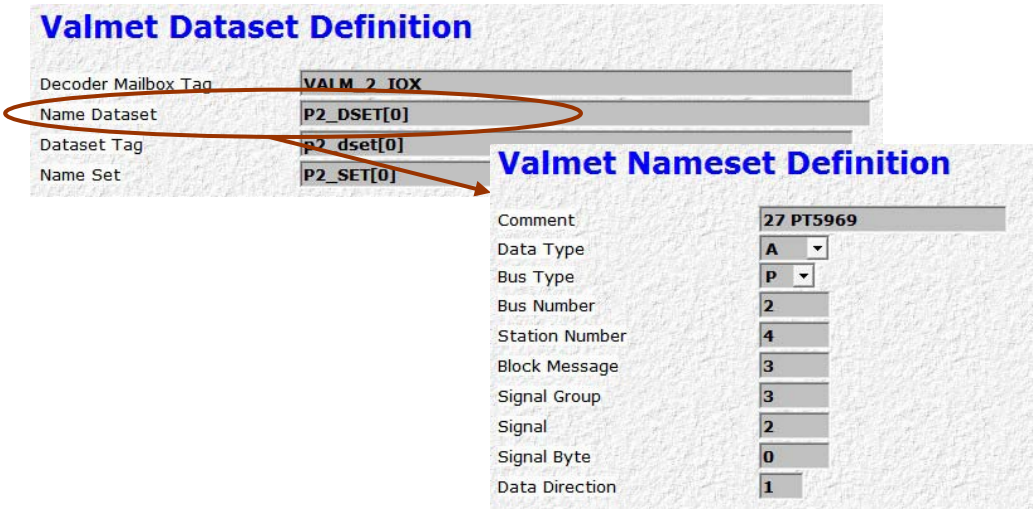


Figure 4.11 Configuration panel VALMET set definition.

The Valmet Nameset definition panel allows the user to specify the actual contents of a datasets for a particular logical device. Different datasets names are selected in the panel *Valmet Dataset definition*. The name of the set is device is displayed in the field Table *Name*, at the bottom of the table. Specify the following information (for details on the meaning the user is also referred to the Valmet Damatic manuals):

Name Set

Text to be filled in by the user for documentation purposes. The text can be used for example to store the corresponding 'tagname' within Valmet or to make it easier to define the panels in the decoder (e.g. by inserting here a sequence number). The text is not used by the protocol driver.

entry ✕ Required
entry type Alphanumeric string of up to 20 characters

Data type

Type definition of the value in the Valmet Damatic computer.

entry ✕ Required
entry type ANALOG refers: A, C, D
description There are 3 types possible: ANALOG (or A or ANA) , COUNTER (or C or COU) , DIGITAL. (or D or DIG).
ANALOG specifies an analog 16 bit value for example the level of a tank.
COUNTER specifies a 32 bit counter value.
DIGITAL specifies to fetch a number of bits (digitals) together with associated status information. The number of bits that is actually fetched depends on the address that is specified. (See further on).

Bus type

The type of bus to access.

entry ✕ Required
entry type Character: P, C
Description Two bus types are possible: PROCESS (or P) and CONTROL (or C). PROCESS specifies to access the process bus. CONTROL specifies to access the control bus.

Bus number

Specifies the number of the bus to access.

entry Required
entry type Numerical value 0..7

Station number
Specifies the number of the station to access.

entry Required
entry type Numerical value 0..15

Block message
Specifies the number of the block to access within the station.

entry Required
entry type Numerical value 0..7

Signal group
Specifies the number of the group to access within the block message.

entry Required
entry type Numerical value 0..3

Signal
Specifies the signal to access within the group.

entry Required
entry type Numerical value 0..3
Description In case a COUNTER value is accessed, the signal number should be 0 or 2.

Signal byte
Specifies (in case of a digital signal only) the byte to access.

entry Required
entry type Numerical value 0..1
Description For non-digital signals this value must be 0.
For digital signals: if 0 is filled the lower byte is accessed., if 1 is specified the higher byte is accessed.

Data direction
Specifies the data direction.

entry Required
entry type Numerical value 0..15
Description In case data from a process station is accessed, a 0 must be specified. Otherwise 1 value greater than 0 must be specified. The number then specifies the control room station to access.

4.7.1 Additional remarks

4.7.1.1 Size restrictions

For an external system to be able to access values in the Valmet Damatic computer, the external system must carry out a two-stage action.

- First, the external system must define the values that need to be accessed. The definition must be sent to the Valmet computer;
- Second, data can be read by accessing the definitions that have been previously sent.

The protocol driver takes care of sending the definition for a dataset to Valmet the first time a dataset is accessed after start-up of the driver of the first after a 'table redefine' is requested (see the description of the Device panel).

Due to protocol limitations, a dataset can access at most 32 values. Counter values are twice as large and are counted double. So one can define in a dataset at most 32 analog/digital values or at most 16 counter values. If counter values and non-counter values are mixed the number of values in a dataset ranges between 16 and 32.

Due to limitations on the size of the definition table in the Valmet computer, each device can define at most 200 values in all the sets that belong to the device. Counter values are twice as large and are counted double. So one can define in all datasets of a device at most 200 analog/digital values or at most 100 counter values. If counter values and non-counter values are mixed the number of values in a device ranges between 100 and 200.

If the user makes a device and/or dataset definition that violates these constraints, the driver will report the error and exit immediately.

4.7.1.2 Errors in set definitions

In case the user defines an Valmet address in a set that does not exist or defines an address that does not match the defined datatype (for example specifying the datatype DIGITAl in the protocol driver while the value is actaully an analog value) results in Valmet returning 2 bytes (or 4 in case of a counter) filled with the hexadecimal value FF. A corresponding converted (!) value will be displayed by FactoryLink.

4.7.1.3 Translator tag definitions

To access the values read from the Valmet device, the user needs to define Decoder panels accordingly. For the Valmet driver some special precautions must be taken. These precautions will be described below.

Due to the nature of the Valmet Datamatic CIS interface it is not possible to use absolute addressing. Relative addressing must be used.

The address field must be filled as follows:

- fill 0 to access the fault code that Valmet always sends back.
- the address of the entry in the Decoder dataset that matches the entry <n> in the corresponding protocol driver dataset is computed from the address of entry <n-1> by adding 1 to the previous address if entry <n-1> is analog or digital and by adding 2 to the previous address if entry <n-1> is a counter.

Analog values

For analog values the Valmet computer sends, apart from the value itself, also status-like information related to the value. To access 'status' information and store it in a digital tag, you must specify the same address as for the analog value itself and use a bit field as follows:

Controller output value

- fill 0 to access the fault code for a controller output value
- fill 1 to access the auto/manual status for a controller output value

Measurement result

- fill 0 to access the O.K. status (value 0) or fault in process connection error (value 1) for a measurement result.
- fill 0 to access the range overflow error (value 1 if overflow).

Digital values

For every digital value defined in the Valmet set, the Valmet computer sends back 8 bits (depending on the signal byte) and, depending on the type of value in Valmet, some status information. To access 'bit' and 'error' information and store it in a digital tag, you must specify the address as explained above and use a bit field as follows (the titles specify the type of value that is accessed in Valmet):

Information of 4 controlled selector switches

- fill for selector <i> (i=0..3) <8+2*i> to access the open (value 0) or closed (value 1) position of the switch.
- fill for selector <i> (i=0..3) <9+2*i> to access whether the switch is in intermediate position (value 1) or not (value 0).
- fill 0 or 1 to access general error information for this area.

Information of 8 switches

- fill for selector <i> (i=0..7) <8+i> to access the open (value 0) or closed (value 1) position of the switch.
- fill 0 or 1 to access general error information for this area.

Information about controller mode

- fill 0 to access whether the output indicates continous control (value 0) or indicates the need to change control (value 1).
- fill 1 to access whether the controller is on tape 1 (value 0) or on tape 2 (value 1).
- fill 2 to access whether the block operates as a controller (value 0) or as a integrator (value 1).
- fill 3 to access whether a computer set value exists (value 1)or not (value 0)
- fill 4 to access whether a remote set value exists (value 1)or not (value 0)
- fill 5 to access whether Damatic controls the setpoint (value 0) or whether the computer controls the setpoint (value 1).
- fill 6 to access whether the controller is in local mode (value 0) or remote mode (value 1).
- fill 7 to access whether the controller is in manual mode (value 0) or auto mode (value 1).

Conversion types

In the decoder you must also specify a conversion to access the requested (analog) value. Fill the conversion type as follows:

To access the analog value itself, specify ANALOG.
To access the lower alarm limit , specify ANALOG.
To access the upper alarm limit , specify ANALOG.
To access the T_I or K_N parameter, specify ANALOG.
To access the K_p or T_D parameter, specify ANALOG.

This page is left blank intentionally.

5 Valmet protocol driver error codes

The error code is returned to the user in a user-defined status tag. These error codes will also be printed with the message of the Valmet protocol driver in the run-time manager. The errors can be generated from different parts of the driver which will be listed here:

Error #	(Serial) communication errors
301	Open function returned OS error code
302	Close function returned OS error code
303	Error retrieving port settings
304	Error setting port settings
305	Memory allocation error, rx or tx buffer
306	Read error serial device
307	Write error serial device
308	Read timeout
309	Error reading time out values for device
310	Error setting time out values for device
311	Invalid mode for deivce
312	Hardware break condition
313	Framing error
314	Hardware IO error
315	Overrun error input buffer
316	Parity error
317	Receive buffer overflow
318	Transmit buffer overflow
319	Error creating semaphore
320	Error releasing semaphore
321	Error closing semaphore
322	Timeout waiting for lock on device
323	OS error locking serial device
324	Error executing break state
325	no pool of communication ports exist
326	Error adding com port to pool
327	no free port could be found
328	Error setting baudrate
329	invalid thread id
330	invalid purging serial buffer
331	error setting signal mask
332	wrong number of data bits
333	wrong number of stop bits
340	modem command error
341	Calling winsock stack failed
342	Invalid winsock version found
343	Creation of socket failed
344	Connection to remote host failed
345	Error sending TCP data
346	Error receiving TCP data
347	Connection needed, but no present on connection

Error #	VALMET driver status / errors
350	Read functionality has been disabled by the user
351	Write functionality has been disabled by the user
352	The com port could not be opened.
353	An incorrect BCC was returned by Valmet device.
354	An invalid type of answer was returned by Valmet device.
355	An invalid data length was returned by Valmet device.
356	An invalid number of words was returned by Valmet device.
357	A time out occurred before a valid or expected message was returned.
358	Extraneous characters followed the checksum byte.
359	The message returned by Valmet had an unexpected composition.
360	A read/write was requested while thedriver was disabled with the disable-tag.

Error #	VALMET driver status / errors
361	A read/write was requested, using the TCP layer, with the TCP-connection disabled.
362	The maximum number of words for the specified dataset is exceeded, reconfigure the dataset and shorten it.
363	Timeout receving an 'ACK' from the Valmet device, read/write request is aborted.
364	Unsupported read or write request, request ignored.
365	Internal error

Error #	FactoryLink errors
401	Internal error, general error code
402	Out of memory, memory allocation error because of a system exhaustion or FactoryLink allocation
403	Unable to map kernel semaphores
404	Can not locate real-time database for given {FLNAME}
405	Can not locate real-time database for given {FLDOMAIN} + {FLUSER}
406	Incorrect function, reserved error code
407	Incorrect argument, bad input parameter passed into function
408	TAG structure received with an invalid offset member
409	TAG structure received with an invalid type or offset member
410	Null pointer assignment
411	Change read/wait function call failed because no changes for given tag array
412	Procedure table full, maximum number of tasks has registered for the given {FLNAME}/{FLDOMAIN}/{FLUSER}
413	Bad procedure name
414	Bad user name
415	Bad option
416	Incorrect checksum
417	No options
418	No key
419	Bad key
420	No port available
421	Port busy
422	Initializing task with same name as one currently running
423	Attempted to unlock the kernel, without holding a lock
424	Unable to attain lock to the kernel
425	Attempted to release a lock that has expired on the kernel
426	Invalid task ID passed into fl_wait()
427	Termination flag for given task ID set to ON
428	Internal error, failed to allocate mailbox queue
429	Internal error, inconsistent queue size
430	Reserve, no tag list
431	Reserved, tag list changed
432	Failure occurred awakening task sleeping on the kernel
433	No signals pending for caller
434	Task awakened by signal from another task
435	Called mailbox function with a tag not f type MAILBOX
436	No mailbox message pending for mailbox tag
437	Can not read a mailbox message targeted for another task
438	Attribute name table full
439	Invalid attribute name
440	Cannot find requested attribute
441	Attempted to start application with same {FLNAME} as one running
442	Unable to map to kernel memory area for {FLNAME}
443	Task protection bit not set for initializing task
444	Attempted to start restricted task from running on a lite system
445	Unable to get or release semaphore
446	Mailbox message space exhausted
450	Float (FLP) is not-a-number
451	Float (FLP) is positive infinity
452	Float (FLP) is negative infinity

Error #	FactoryLink errors
501	FLDTP general failure indicator
502	FLDTP invalid argument passed dtp function
503	FLDTP invalid tag passed to dtp function
504	FLDTP memory allocation failure
505	FLDTP can not find requested object
506	FLDTP illegal call to procedure at given time. Some dtp functions are not re-entrant
507	FLDTP kernel function call failed.
508	FLDTP tag of unknown type passed into function
509	FLDTP no tag changes are pending

Error #	MCI errors
550	Bad message type
551	Message with dataset control tag not found in queue
552	No messages available to query
553	Bad receive mailbox tag
554	Bad mailbox send tag
555	Bad dataset control tag
556	Message cannot be adjusted
557	Operation to big for variable
558	Unknown boundary
559	Function not supported
560	No message for this index present
561	The remote dataset Is not defined on this system
562	The received dataset was not registered
563	The message is not queued
564	Message is rejected due error in the remote MCI
565	Illegal method of addressing bits on bit boundary
566	Element cannot be written
567	Invalid buffer specified
568	Block write function impossible
569	Maximum number of messages in MBX reached
570	No memory left
571	Error registering standard dataset
572	Error writing message in pipe
573	Not supported MCI message
574	Error creating pipe
575	Error starting threat
576	Error server connecting to pipe
577	Error child connecting to pipe
578	Error reading the pipe
579	Error number of bytes read from pipe
580	Error writing into the pipe
581	Error number of bytes written into the pipe
582	Error reading dataset from the mailbox
583	No communication buffers assigned
584	Error decrementing semaphore
585	Error writing to pipe, no space left
586	Error querying no. of messages for translator
587	Max. No. of messages in translator MBX reached

This page is left blank intentionally.

6 Valmet protocol driver error messages

If an error condition occurs in the protocol driver task during run-time mode, a message to that effect will appear on the runtime manager graphics screen to the right of "Valmet". The error messages that may be displayed are as follows:

Can't open CT file
The protocol driver task was unable to open the configuration table file, generally because it does not exist. This is a fatal error.

No triggers defined
The protocol driver configuration table file does not contain an expected trigger. This is a fatal error.

No Datasets defined
The protocol driver configuration table file does not contain a valid dataset definition. This is a fatal error.

No Decoders defined
The protocol driver configuration table file does not contain a valid decoder definition. This is a fatal error.

Unknown decoder table %s record %d type %d
The protocol driver configuration table file contains an unknown decoder mailbox tag, for the specified table, record number and type of dataset. This is a fatal error.

Reading CT index
An error occurred during reading a Configuration Table index, normally this means the CT-file is corrupted. This is a fatal error.

Reading CT header
An error occurred during reading a Configuration Table header, normally this means the CT-file is corrupted. This is a fatal error.

Reading CT record
An error occurred during reading a Configuration Table record, normally this means the CT-file is corrupted. This is a fatal error.

Out of RAM
There is not enough RAM memory to load the complete configuration and/or task.

Invalid tag number
The protocol driver encountered an invalid TAG number. This is a fatal error.

Bad tag %s %s %d
The protocol driver encountered an incorrect TAG definition. The device, functionality and the error number are displayed.

Reading data %s %s, %d
The protocol driver encountered an error reading a TAG. The device, functionality and the error number are displayed.

Mbx query %s mbx, %d
The protocol driver encountered an error reading a mailbox TAG. The name of the TAG and the error number are displayed.

Change read %s %s, %d
The protocol driver encountered an error during a change read of a TAG. The device, functionality and the error number are displayed.

Forced write %s, %d

The protocol driver encountered an error during forced writing a TAG. The device and the error number are displayed.

MCI init %d
MCI initialisation error, internal error of the protocol driver task. Specified is the error number.

MCI maximum msg in Decoder MBX %d reached
MCI error, the maximum number of allowed messages in a decoder mailbox tag is reached. Specified is the record number of the decoder mailbox.

MCI initialising PD queue %d
MCI initialisation error, internal error of the protocol driver task. Specified is the error number.

MCI receive from PD mailbox %d
MCI error, internal error of the protocol driver task. Specified is the error number.

MCI write to send mailbox %d
MCI error, internal error of the protocol driver task. Specified is the error number.

MCI search control tag %d,%d device %s %d
MCI error, internal error of the protocol driver task, a previously registrated command in the mailbox could not be found. Specified are the dataset control tag, device name and the error number.

MCI get number of messages %s
MCI error, an error occurred during reading a mailbox tag element. Specified is the error number.

MCI identify dataset by number not supported %d
MCI error, decoder tried to identify a dataset by a number. This is not supported in this version of the protocol driver. Specified is the dataset number.

MCI received msg couldn't be converted to dataset
MCI error, a received message in the mailbox tag element is not recognised as a valid read or write command.

MCI msg adjust %s on ds %d:%d: %d
MCI error, a received message in the mailbox tag element could not be interpreted as a valid read or write command for the external device. Specified are the device name, dataset control tag and the error number.

MCI unknown msg type %d
MCI error, internal error of the protocol driver task. Specified is the number of an unrecognised command.

MCI Bad MSG Control TAG table %s record %d type %d
MCI error, internal error of the protocol driver task. Specified are table number, record number and type number.

MCI Duplicate MSG Ctrl TAG table %s record %d type %d
MCI error, internal error of the protocol driver task. Specified are table number, record number and type number.

MCI Bad Mailbox TAG table %s record %d type %d
MCI error, internal error of the protocol driver task. Specified are table number, record number and type number.

MCI DS for %s function not found, MsgCtrl %d
MCI error, internal error of the protocol driver task, missing dataset for the task. Specified are function and the message control TAG.

Com init comm layer %d

An error occurred during initialising the communication layer Specified is the error number.

Device %s. Error while communicating %s

Status message for the specified device and function. The meaning of the error number can be found in the previous appendix.

Com building device

An error occurred during building the communication datablock for device. This is a fatal error.

Internal error

Internal error of the protocol driver task, contact RLD Automation for support.

Too many entries in set %s. Number is %d.

The set with the given name violated the size restrictions for a set. The number of entries in the set is given (where counters count double.)

Too many entries in device %s. Number is %d.

The device with the given name violated the size restrictions for a device. The number of entries in the device is given (where counters count double.)

Internal error

Internal error of the protocol driver task, contact RLD Automation for support.

