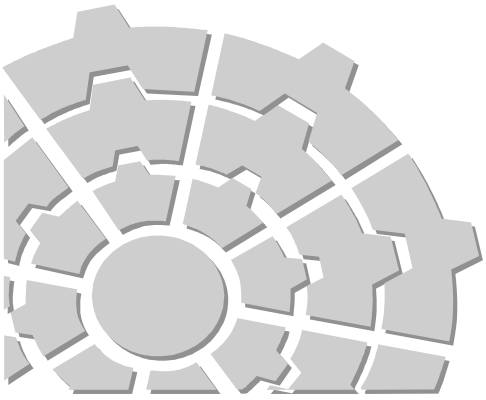
DeltaLink SAPI S7
Simple Application Programmers Interface software for FactoryLink

User Manual



Version 1

Printed june 26TH, 1998

Table of Contents

1. INTRODUCTION 4

1.1. SCOPE OF THIS DOCUMENT 4

2. CONTENTS OF SHIPMENT 6

3. INSTALLATION 8

3.1. INSTALLATION OF THE FACTORYLINK SOFTWARE 8
3.2. INSTALLATION OF THE PROTECTION 11
3.2.1. The DeltaLink option file 11
3.2.2. Demo installation 11

4. PRINCIPLE 12

4.1. THE IOXLATOR RAPD DRIVER PRINCIPLE 12
4.2. H1 COMMUNICATION 13
4.2.1. Datasets 13
4.2.2. Virtual connection 13
4.2.3. Block read and block write 13
4.2.4. Exception write 14
4.2.5. Encoded write 14
4.2.6. Unsolicited receive 14
4.3. FACTORYLINK DOMAIN SELECTION 15

5. CONFIGURATION TABLES 16

5.1. MAILBOX DEFINITION 17
5.2. IOXLATOR DEFINITION 18
5.3. DEVICE DEFINITION 19
5.4. DATASET DEFINITION 21

APPENDIX A. THE DELTA.OPT FILE 23

APPENDIX B. COMMAND LINE PARAMETERS 25

APPENDIX C. SAPI ERROR CODES 26

APPENDIX D. MESSAGES 29

E. INSTALLATION OF CP1413 CARD, CP-PG INTERFACE 32

E.1 CONFIGURATION 34

E.2 CONFIGURING THE TRACE DEPTH OF SAPI 34

E.3 PLC SOFTWARE EXAMPLE 35

APPENDIX F. SIEMENS S7-PLC 35

F.1 S7 H1 PROTOCOL 35

F.2 ENCODED WRITE HANDLING 39

F.4 PLC SOFTWARE EXAMPLE 40

Deltalink SAPI_S7

Provides the following features:

- ☑ Supports the S7-400 PLC range.
- ☑ Supported network types: Profibus, Industrial Ethernet and TCP/IP.
- ☑ Full asynchronic communication on Read, Write and Receive.
- ☑ Selectable boards. Makes it possible to have more then one board in the same PC, and even switch communication from one board to another.
- ☑ Full control over communication.
- ☑ Enable tags on all functions per PLC.
- ☑ Ease of use. Just specify which areas to Read, Write or Receive.

Supported Platforms and FactoryLink Versions:

Platform	FL Version	Part No.	Hardware
Windows NT	6.x.x	FLNTI-SAPI	Depends on choice of network type
Windows 95	6.x.x	FLWIN-SAPI	Depends on choice of network type

Third Party Software / Hardware Dependencies:

- ☑ Siemens communication board, choice depends on network type: CP1413, CP1416 for Ethernet or TCP/IP, CP5412, CP5613 for Profibus.
- ☑ Programmable Logic controller of type Siemens S7 400 series. PLC's must be equipped with a communicaiton card, eg CP143 (Ethernet).

Supported Protocols:

- ☑ SAPI S7 protocol

1. Introduction

Thank you for buying this driver! We hope you will enjoy using this product.

1.1. Scope of this document

This manual is written for a technician who is familiar with both the FactoryLink® IV software and the Siemens S7 or Siemens Programmable Logic Controllers (PLC's). This document can be used both as a training manual as well as a reference manual.

> ***Note*** *Please check the contents of the shipment with the list as described in the next chapter.*

The first section of this manual deals with the installation of hardware and software in your FactoryLink workstation. This part is split into a platform independent and a platform specific part. Please read carefully through both parts to make sure both hardware and software are installed correctly.

The second part explains the operation principles of the communication with the Siemens PLC and the IOXLATOR. Here all terms and definitions are explained to the reader. It explains terms like "Dataset" and "Virtual Connections'. This part should be read by both the PLC programmer and the FactoryLink programmer to make sure that the optimum performance can be achieved.

The third part explains the exact tables associated with this driver. This part is useful only to FactoryLink programmers and can be used as a reference. This part is also an example of how to use this driver with the Ioxlator. It shows the entries made for the pre-configured demo which comes with this package. The demo program can be used to check if the communication is working without making a complete application.

The last part are the appendices which contain summarised data.

DeltaLink bv, Pioenroosstraat 26, 5241AB Rosmalen, The Netherlands,

tel + 31 73 523 1234
fax +31 73 523 1222

This page is left blank intentionally.

2. Contents of shipment

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

This package includes the following

1 diskette labelled "DeltaLink Sapi S7 Protocol Driver"
Files \AC\Sapi_s7.AC
\BIN\SET_PARM.EXE
\BIN\NTIXTIB.DLL
\CTGEN\Sapi_S7.CTG
\HELP\DE\Sapi_S7.HLP
\HELP\EN\Sapi_S7.HLP
\HELP\FR\Sapi_S7.HLP
\KEY\DE\Sapi_S7.KEY
\KEY\EN\Sapi_S7.KEY
\KEY\FR\Sapi_S7.KEY
\MPS\EN\Sapi_S7.MPS
\MSG\DE\Sapi_S7.TXT
\MSG\DE\Sapi_ac.TXT
\MSG\EN\Sapi_S7.TXT
\MSG\EN\Sapi_AC.TXT
\MSG\FR\Sapi_S7.TXT
\MSG\FR\Sapi_AC.TXT
\OPT\DELTA.OPT
\PLC\sapiawl.zip
\INSTALL.BAT
\INST_SEQ.EXE
\FLBUILD.ID
\FLXMEDIA
\UPDATE.EXE
\INSTALL.$$$

❷ 1 DeltaLink authorization sequence for a SAPI_S7 option.

❸ This manual (Which seems to be present).

You should also have

- ☑ A IMX based Ioxlator e.g. DeltaLink Ioxlator.
- ☑ The correct hardware and software for the OSI Ethernet stack on your platform. (See also chapter 3)
- ☑ Siemens or S7-400 PLC with CP443 communication board and software to program both the CPU and the CP boards.
- ☑ An Ethernet-network with at least both the PLC and the FactoryLink station connected to it.
- ☑ a siemens CP1413 communication processor for the pc

This page is left blank intentionally.

3. Installation

3.1. Installation of the FactoryLink software

To install the FactoryLink task and its related tables please follow the following steps.

Before installing

Before installing the driver on the system, FactoryLink must have been installed error free. It is very important that all the environment settings are made for the FactoryLink system such as the *FLINK*, *FLOPT* etc.

First

Copy the files from the "DeltaLink SAPI_S7" media to the appropriate directory. This will be done automatically by running the install utility placed on the installation media. The installation procedure differs for W95-NT systems.

For W95-NT systems follow the next procedure

```
a\↵
install↵
```

Second

After you installed the software you need to activate the tables in the FactoryLink Configuration Manager (FLCM). The installation automatically appends the *sapi_s7.ac* entry into the *{FLINK}/ac/titles* file[1]. The place of this entry is also the place where the option appears in the FLCM Main Menu. Therefore check the validity of the entry and move it to the place where you want to appear it in the Configuration Manager. The entry must match the sapi_s7 entry in the following table

```
file {FLINK}/ac/titles

...
windhdr.ac
loxlator.ac
sapi_s7.ac
spool.ac
...
```

Third

To make sure all the Configuration Tables (CT's) are generated after a change, the install utility automatically adds the sinec_h1 entry at the end of the *{FLINK}/ctgen/ctlist* file. The place of this entry is not important. Check if this entry has the same format as in the next table.

```
file {FLINK}/ctgen/ctlist

...
recipe rcphdr rcpovr
loxlator loxlatorm loxlatorp loxlatord loxlatort
sapi_s7 sapis7d sapis7p sapis7x sapis7m
iml imltags imltrig
...
```

Fourth

[1] {FLINK} is the working directory for the FactoryLink programs.

To enable the help functionality for the Siemes Sapi S7 Protocol Driver tables in the Configuration Manager, the installation utility reindexes the 'help-index' for the Configuration Manager. If desired reindexing of the 'help-index' can be started from the command line prompt.

```
mkhelp↵
```

Fifth

The FactoryLink Configuration Manager uses a map file, {FLINK}/ac/ac2ct.map, to be aware of the different configuration tables which can be located behind one entry in the main menu. Upon startup, the Configuration Manager reads the map file instead of all the table configuration files, mainly because reading all these files at once takes too much time. An account manger is present to update this conversion file, and can be started from the command line.

```
acctmgr  -c -d -t{FLINK}/ac/titles↵
```

Sixth

The protocol driver must, with 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*
Sapi_s7	***Siemens Sapi s7 Protocol Driver***	***bin/sapi_s7***

The Task Name and name of the executable file are fixed and should not be altered by the user.

This completes the installation of the FactoryLink (software) parts.

3.2. Installation of the protection

The Siemens Sapi S7 protocol driver is protected via the DeltaLink option file. This file contains authorization sequence codes for DeltaLink modules. The protection is linked to the serial number of the FactoryLink package.

3.2.1. The DeltaLink option file

The installation media of the Sapi S7 driver contains an option file, named ‘delta.opt’, in the 'opt' directory. This file contains the unique authorization sequence which enables the Sapi S7 driver to run. The install utility automatically copies the authorization sequence into the {FLOPT}/delta.opt file. It is also possible to enter the authorization sequence manually into the {FLOPT}/delta.opt file. For more information on the delta.opt file refer to *Appendix B*.

Note that the task will only run on the FactoryLink system with the same serial number. The *delta.opt* file on the installation diskette contains, for reference, the serial number of FactoryLink.

3.2.2. Demo installation

It is possible to install the Sapi S7 driver without an authorization code. This will be done from a normal installation media. In this the task will start up but only runs in so called 'demo' mode. This means that the driver runs only for a limited period of time (five hours). After this period has expired the task will shutdown and can not be restarted before the complete FactoryLink system has been restarted.

After installation of this demo version an authorization code can be ordered and installed which enables the task to run without the time and restart limitation. The authorization code must be entered manually in the {FLOPT}/delta.opt file. For more information on entering the authorization code in the delta.opt file refer to *Appendix B*.

The limitations of a demo version of the Siemens Sapi S7 driver are

- five hours of consecutive run-time
- not restartable (FactoryLink must be restarted)

4. Principle

4.1. The Ioxlator RAPD 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 Intertask Mail Exchange Standard or IMX, which defines a way for a protocol driver task to communicate with an I/O Translator task (e.g. Ioxlator, high speed logger). The RAPD system consists of a protocol driver which communicates with external devices (RTUS, PLCs, etc.) and a translator which 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 communications 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.
The protocol driver and the IOX communicate with one another via FactoryLink mailbox tags, according to the IMX standard. Every task (translator and protocol drivers) 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 IMX standard is especially designed for the following situation. To use one Ioxlator and several protocol drivers. For example the Ioxlator together with the Sapi S7 protocol 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.

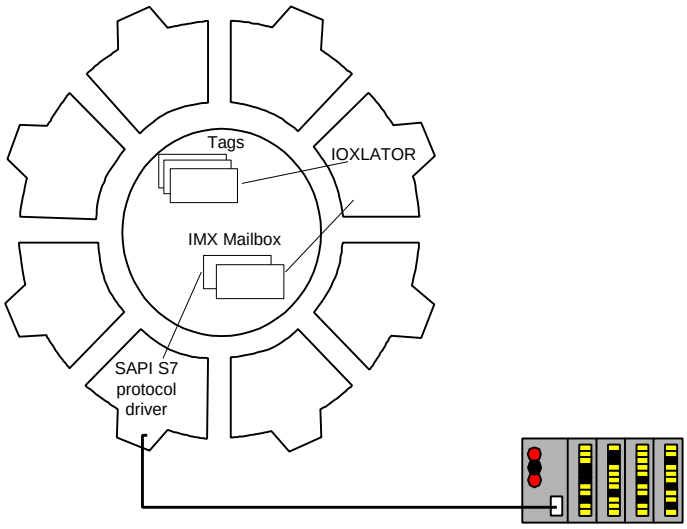


Figure 4.1.1 The RAPD principle.

4.2. H1 Communication

4.2.1. Datasets

A dataset is a (contiguous) area of data in the PLC. The complete area can be read or written to, with one command (trigger). Except actions on the complete area, the protocol driver is also capable of addressing specific elements (or a group of elements) in the data area. The Ioxlator takes care of mapping the PLC-data to/from FactoryLink Tags (including conversions).
A dataset consist out basic elements, which are all of the same size, for example all words or all bytes. This size corresponds with the size definition of the data area in the PLC and is called the boundary of a dataset. The boundaries for the different data areas used with the Sapi S7 driver are listed in the tables below.

Siemens S7	
PLC data type	Boundary
Data block	Byte
Flags	Byte
Input	Byte
Output	Byte

4.2.2. Virtual connection

Data exchange between a FactoryLink workstation and a PLC is performed by using the *read*- and *write*-services of e.g. the Siemens CP1413 communication processor. The protocol driver task uses the *read*-service to read (or fetch) data from the PLC, and it uses the *write*-service to write data to the PLC. Both actions can be initiated by the driver task, also the PLC may use the *write*-service for sending data to the FactoryLink station, the unsolicited receive for the protocol driver.

Every *read/write/receive*-service is associated with a so called Virtual Circuit (VC). A Virtual Circuit is a bi-directional point-to-point communication link through the network. Each Virtual Circuit requires a Transport Service Access Point Identifier (TSAP-ID). To set-up a Virtual Connection between two stations, the following information is needed for each station.

- Local Ethernet address.
- Local TSAP.
- Remote Ethernet address.
- Remote TSAP.

These are configured in the COML S7 interface from SIMATIC NET.
The Sapi S7 protocol 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.

Read/Write Service	Supported
Block read	Yes
Block write	Yes
Exception write	Yes
Encoded write	Yes
Unsolicited Receive	Yes

4.2.3. Block read and block write

The block read and block write commands are performed on a contiguous block of data in the PLC, the dataset. Both commands are initiated by the FactoryLink station. The Ioxlator sends requests to the Sapi S7 protocol driver for reading or writing an entire dataset. For configuring the Ioxlator refer to the manual of the Ioxlator.

4.2.4. Exception write

An exception write is initiated by the FactoryLink workstation, and writes data to the PLC. Not a complete dataset but one element from the dataset is written to the PLC. The protocol driver will receive a request for an exception write from the Ioxlator. The first action the driver will take is to check if the write can be accomplished with one write or must be read first.

In the exception data element size is smaller than the size of the PLC data type element (boundary) the data must be read first, patched with exception data and written back. If this is not done, data will be unintentionally overwritten in the PLC. Whether this is the depends on the data type of the dataset.

For example a Siemens S7 PLC if a data left (DL) in a Data Block (DB) has to be set then the specific data word has to be read first because the smallest element in a Data Block is a word. The DL has to be patched into the word and written back.

In the above mentioned example it is possible, that the device changes the contents of the word, after the protocol driver has read the word, but before the new value is written. This means that after the write of the protocol driver the content change of the PLC is lost. For applications which require that single elements can be accessed both by the PLC and the protocol driver, an encoded write can be used. The encoded write will be discussed in the next section.

4.2.5. Encoded write

The encoded write function is almost similar to the exception write in the fact that both functions write single data elements. The difference is that the exception write accesses directly the destination address in the PLC, whereas the encoded write composes an encoded write command and writes this command to a location in a certain data block. The PLC program reads and decodes this encoded write command and accesses the actual destination location.

The advantage of this method is that for every element to write only one write command is needed because the PLC program does the actual operation on the data element (no reading before writing). Another advantage is that the PLC program can control all encoded writes because they all are written to one location in the PLC.

The location where the encoded write command will be placed is part of the definition of a logical device, a detailed description is in the next chapter.

4.2.6. Unsolicited receive

In of an unsolicited receive the PLC sends a dataset to the FactoryLink workstation, without a specific request from that workstation. This means that the action is initiated by the PLC. The protocol driver will check if the received data is configured in one of the datasets. If this is the , the data will be sent with the dataset information to the Ioxlator which converts and updates the data. If no dataset is specified regarding to the received data, it will be discarded.

The unsolicited receive function is a very powerful functionality for fast communication because no request has to be placed by the protocol driver. This way less overhead is involved in communication. Data which alters unpredictably such as alarms are very suitable for the unsolicited receive function. Some extra programming effort has to be done in the PLC because the PLC is in charge of sending the data.

4.3. FactoryLink domain selection

The standard domain for the Siemens Sapi S7 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 Ioxlator must be in the same domain (either SHARED or USER).

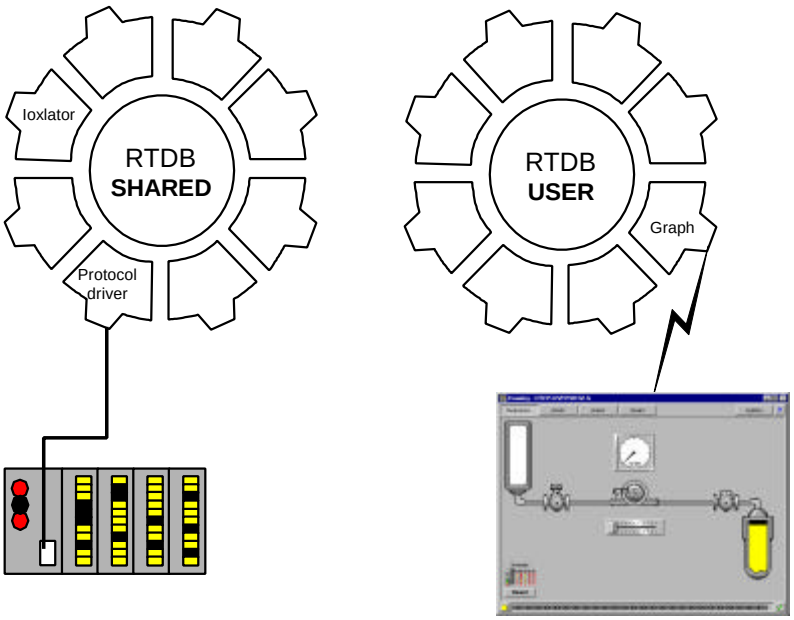
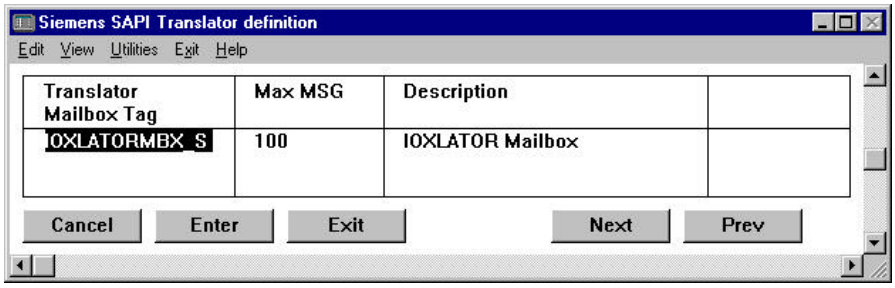


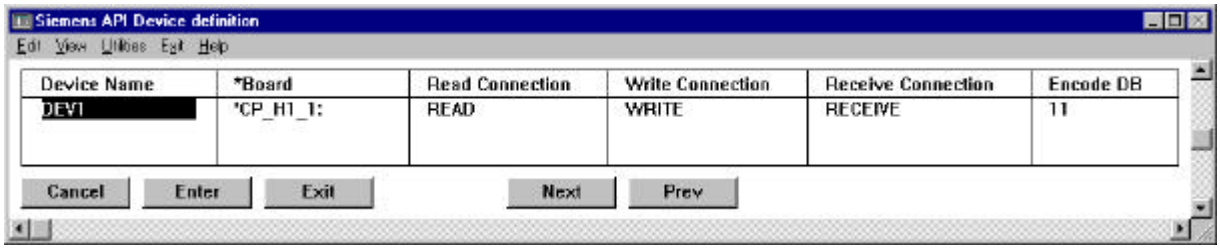
Figure 4.3.1 Standard domain selection.

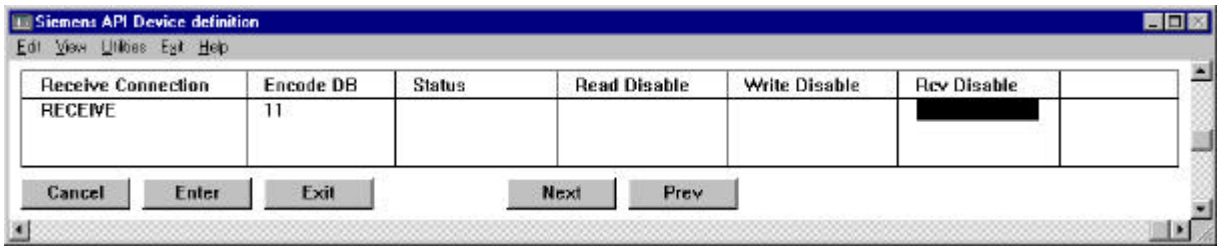
5. Configuration tables

In the Configuration Manager Main Menu, select **DeltaLink Sapi S7 Protocol Driver**. Four tables appear, with the titles of all panels visible for direct access. To access a specific panel position the cursor on a visible area and press the left mouse-button, or use the Next/Prev buttons .

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







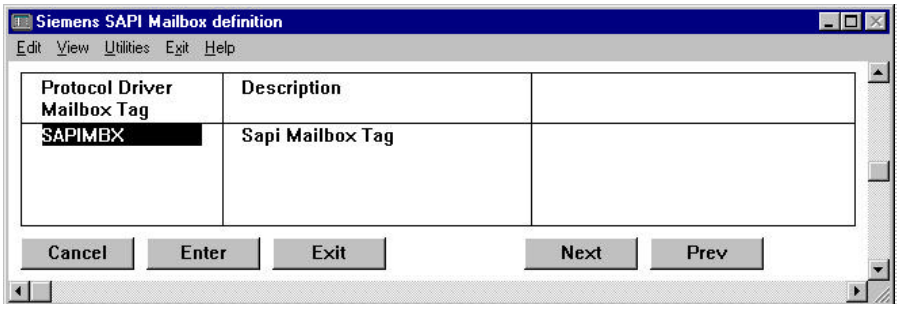
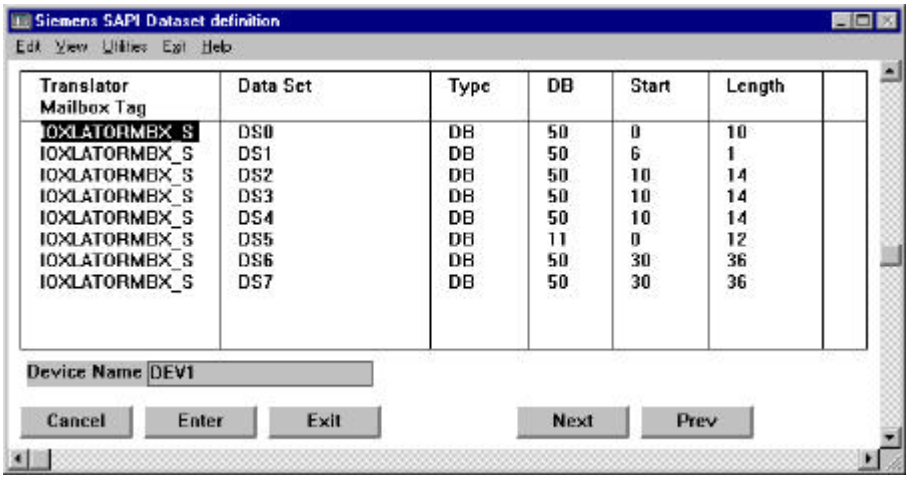


Figure 5.0.1 Siemens Sapi S7 Configuration panels.



5.1. Mailbox definition

From the display of all the panels, select the *DeltaLink SAPI Mailbox Definition* panel.

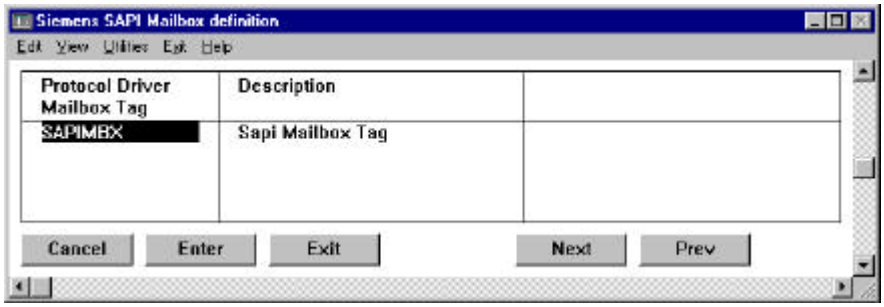


Figure 5.1.1 DeltaLink H1 Mailbox Definition.

The Siemens H1 mailbox definition panel allows the user to initialise one Mailbox Tag for the Sapi S7 protocol driver. Only one Mailbox is needed for full performance of IMX. Specify the following information.

- **Protocol Driver Mailbox Tag**
  Tag name of the Sapi S7 protocol mailbox element that the developer defines, to be referenced by a Ioxlator task (**N.B.** IMX must be supported by the Ioxlator task). The Ioxlator task uses this mailbox to send requests to the Sapi S7 protocol driver.

  entry Required.
  entry type Standard FactoryLink tag name.
  valid entry MAILBOX.

- **Local Tsap**
  Local TSAP identifier of the FactoryLink workstation. The entry made here must be identical with the remote TSAP identifier of the connected device(s).

  entry Required.
  entry type Alphanumeric string.
  valid type String of up to 8 characters.

- **Description**
  Description of the Sapi S7 protocol driver mailbox element defined by the developer.
  valid entry Output only.

5.2. Ioxlator definition

From the display of all the panels, select the *DeltaLink H1 Ioxlator definition* panel.

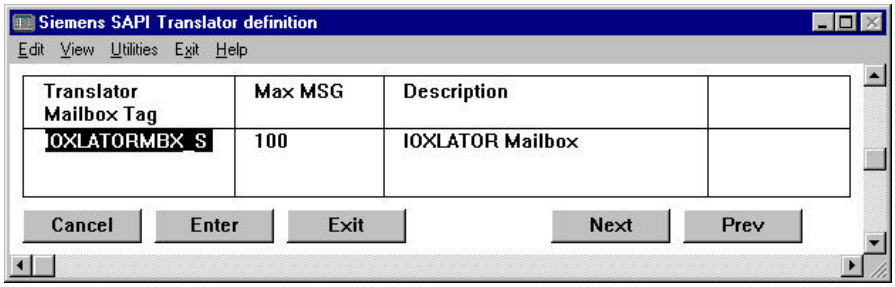


Figure 5.2.1 DeltaLink Ioxlator Definition.

The Sapi S7 Ioxlator Definition panel allows the user to specify one or more Ioxlator(s). Every Ioxlator mailbox tag specifies a different Ioxlator. Only one mailbox tag, for every Ioxlator, is needed for full performance of IMX. Specify the following information.

- **Ioxlator Mailbox Tag**
  Tag name of a Ioxlator mailbox element, to be referenced by the Sapi S7 protocol driver task (**N.B.** IMX must be supported by the Ioxlator task). The Ioxlator task uses this mailbox to receive data from the Sapi S7 protocol driver.

  entry Required.
  entry type Standard FactoryLink tag name.
  valid entry 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 / Default 100.
  entry type Decimal number.
  valid entry Positive integer 1 .. 9999.

- **Description**
  Description of the Ioxlator mailbox element defined by the developer.

  valid entry Output only.

5.3. Device definition

From the display of all the panels, select the *DeltaLink SAPI_S7 Device Definition* panel.

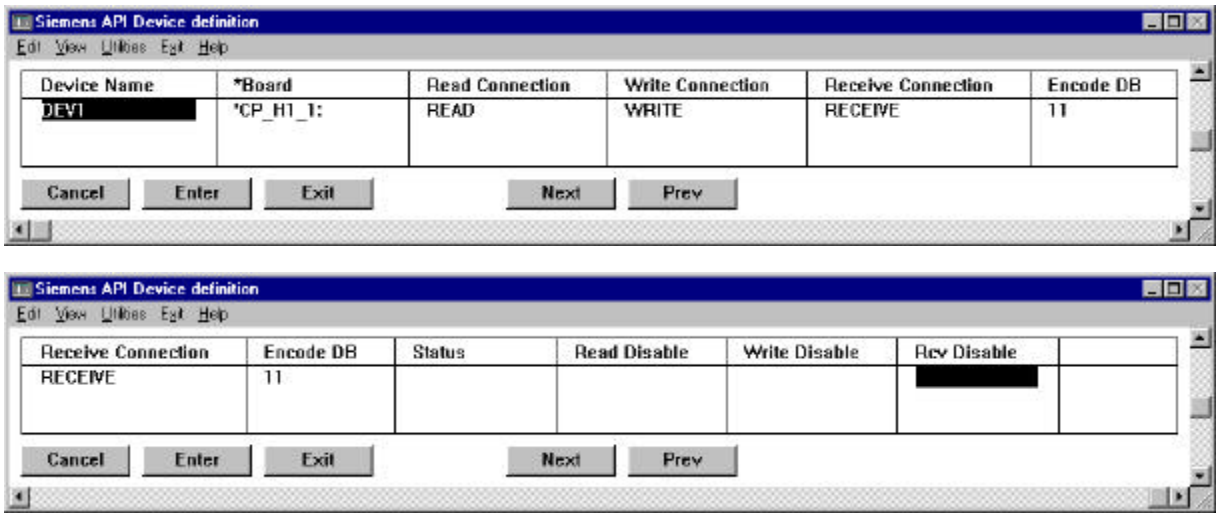


Figure 5.3.1 DeltaLink SAPI_S7 Device Definition.

The DeltaLink S7 Device definition panel allows the user to specify logical devices, for the communication. Different logical device names may be used for one communication device (the same Ethernet address). If the same Ethernet address is used make sure that the next entries are not identical Device Name, Remote TSAPs. Specify the following information.

- **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 *DeltaLink S7 Dataset Definition*

  entry Required.
  entry type Alphanumeric string.
  valid type String of up to 16 characters.

- ***Board**
  Logical name assigned by the user to represent a particular board. If preceded by a " ' " then the board will be regarded as string else the name is a tag name. If one has chosen a name in the Simatic Net software one should use this name in the control panel and in this case this name is ending with the character ":" .

  entry Required.
  entry type Alphanumeric string or tag name
  valid type String of up to 16 characters.

- **Encode DB**
  Data Block (DB) number for encoded write commands. Coded information is placed in this DB, encoding is performed by the PLC. A standard function block (FB10) is provided by DeltaLink for decode actions inside the PLC, for specific information see also *Appendix F* and *Appendix G*.

  entry Optional.
  entry type Decimal number.
  valid entry Positive integer 1 .255.

- **Status Tag**
  Real-time database element written to by the Sapi S7 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. A value of zero indicates no error (success), any other values represents an error. (appendix C)

entry	Optional.
entry type	Standard FactoryLink tag name.
valid entry	ANALOG.

- **Read disable**

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

entry	Optional.
entry type	Standard FactoryLink tag name.
valid entry	DIGITAL.

- **Write disable**

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

entry	Optional.
entry type	Standard FactoryLink tag name.
valid entry	DIGITAL.

- **Receive disable**

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

entry	Optional.
entry type	Standard FactoryLink tag name.
valid entry	DIGITAL.

5.4. Dataset definition

From the display of all the panels, select the *DeltaLink SAPI Dataset definition* panel.
The Sapi S7 Dataset Definition panel allows the user to specify datasets for a particular logical device. Different logical device names are selected in the panel *DeltaLink SAPI Device Definition*. The name of the logical device is displayed in the field *Device Name*, at the bottom of the table. Specify the following information.

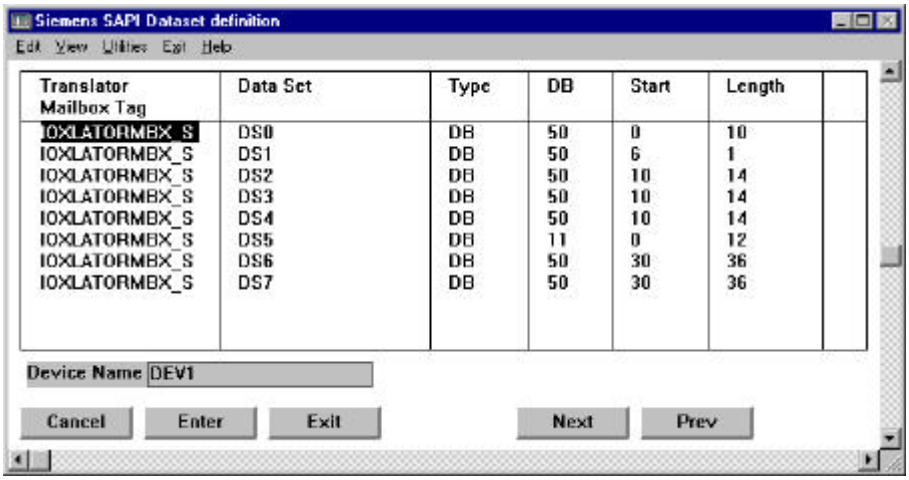


Figure 5.4.1 DeltaLink SAPI Dataset definition.

- **Ioxlator Mailbox Tag**
  Tag name of a Ioxlator mailbox element, to be referenced by the Sapi S7 protocol driver task (**N.B.** IMX must be supported by the Ioxlator task). The Ioxlator task uses this mailbox to receive data from the Sapi S7 protocol driver.

  entry Required.
  entry type Standard FactoryLink tag name.
  valid entry MAILBOX.

- **Dataset**
  Tag name representing a (unique) logical name for a dataset. The tag is used internally in the Ioxlator and protocol driver for activating a dataset read from the external device. A dataset is referenced, internally by the Ioxlator and the protocol driver, by the tag name. Therefore the tag specified in this field must be unique.

  entry Required.
  entry type Standard FactoryLink tag name.
  valid entry DIGITAL.

- **Type**

  Type definition of the data area in the logical device (PLC). Note that exception write behaviour depends on the type of data area. To know how an exception write is executed, one should regard the boundary of the data area and the element size involved with the exception write.

  entry Required.
  entry type Alphanumeric string.
  valid entries DB, FB, IB, OB, CW, TW, SW, PB, EB, DX.
  Description:

Siemens S7 type	
Data type entry	Description
DB	Data Block
FB	Flag byte
IB	Input Byte
OB	Output byte

- **DB**

  Data block (DB), External Data block (DE) or Data block eXtended (DX) number. Only used if the *Type* is DB, DE or DX.
  In a TI data type has been specified this number will be used for addressing pages of 64Kb within the data type. Usually this will be set to zero. For example, to address variable with address 65536 in V memory specify DBNR 1 and start address 1.

  entry Optional.
  entry type Decimal number.
  valid entry Positive integer 1 ..255.

- **Start**

  Start address number of the dataset.

  entry Required / Default 0.
  entry type Decimal number.
  valid entry Positive integer 0 ..2048.

- **Length**

  Length of the dataset in bytes or words, depending on the type of the dataset. For the types DB, DE and DX the length is expressed in words, the data boundary is words. For all other types the data boundary is byte, which means that the length is expressed in bytes. Maximum length depends on the type of data set and PLC type, for further details see the Siemens manuals.

  entry Required / Default 1.
  entry type Decimal number.
  valid entry Positive integer 1 ..2048.

Appendix A. The delta.opt file

In order to run the task permanently with full functionality an authorisation sequence for the task option is needed. This appendix describes the authorisation sequence together with the option file.

The authorisation sequence of every independent DeltaLink module must reside in the {FLOPT}/delta.opt file in order to take effect. In the task has been ordered with an authorisation sequence then this sequence resides on the install diskette (in the /opt/delta.opt file) and will be automatically added to the {FLOPT}/delta.opt file with the installation. In the task has been previously installed from a demo diskette and the authorisation sequence has been purchased later then the sequence must be added manually to the {FLOPT}/delta.opt file.

The format of the delta.opt file consists of two parts. The first part is the comment header. This part remains always at the beginning of the file. Every line of the comment part starts with an '*' character. The second part contains the authorisation sequences of every independent DeltaLink module. Every line must hold one sequence code and must apply to a strict format.

An example of the option file looks like this

Protection file *delta.opt*

```
*
* Copyright 1998 DeltaLink bv. All Rights Reserved
*
* DeltaLink bv
* Pioenroosstraat 26
* NL-5241 AB Rosmalen
* The Netherlands
* Tel (int) 31 73 5231234
* Fax (int) 31 73 5231222
*
*
* FactoryLink Serial Number 12345OS2
*
* DeltaLink module option
*
* MODULE = taskname of always 8 characters with trailing spaces
* .      = <space>
* X      = authorisation code supplied by DeltaLink
*
* MODULE..XXXX.XXXX.XXXX.XXXX.XXXX.XXXX.XXXX.XXXX.XXXX
SAPI__S7 2D6F E047 2534 036E 1215 EC80 92F8 1EFC C4C0
```

In a full installation diskette with authorisation diskette has been ordered then the header in the option file on the diskette contains the serial number of FactoryLink. The task will run only on the FactoryLink package with this serial number. If the serial number is not listed in the {FLOPT}/delta.opt file due to a previous demo installation then this number can be added in the header of this file.

The authorisation code must exactly match the format as listed in the header. If this is not the the module will not recognise the authorisation sequence and start up in demo mode.

The format of an authorisation sequence line is as follows

MODULE<s><s>XXXX<s>XXXX<s>XXXX<s>XXXX<s>XXXX<s>XXXX<s>XXXX<s>XXXX<s>XXXX<CR><LF>

The **MODULE** field contains the DeltaLink module name in this SAPI__S7. This field must always be 8 characters long. If the module name is shorter then 8 characters then the name must be filled out with spaces to 8 characters.

After the MODULE field one space must be entered.

After the space field 9 records must be specified with the authorisation code. One record is build up of one leading space (ASCII 0x20) and four sequence codes. The sequence codes must be entered exactly as specified by DeltaLink.
After the authorisation code records a carriage return (ASCII 0x13) and linefeed (ASCII 0x10) must follow.

There may be no empty lines between the specification of more than one module. To add an authorisation sequence a normal editor can be used. If all modules with the right authorisation codes are specified according to the format described above then the modules will start with full functionality.

Appendix B. Command line parameters

The protocol driver accepts several command line parameters, these can be configured with the configuration manager in the 'System Configuration' table, column 'Program Arguments'. An argument consists out of first a minus sign ('-'), followed by the a letter specifying the option. After the letter an optional number can be present, if this is supported by the option.

Option	Description
-d*n*	Debug option, the level of debug information is set with the number *n*. The range of this number is from 1 until 4. If no number is specified the default level will be 1. The debug output will be visible in the 'window' of the protocol driver, or 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/sinec_h1.log
-LS*n*	Local Station id with number *n*, needed if the Ioxlator and protocol driver reside on different nodes (or FactoryLink workstations). This must be an unique number for the network.
-s*n*	The stack will be started and stopped automatically. The stack will be started during start-up of the driver, and stopped when the driver is stopping. The 'n' represents the number of seconds the driver will wait after it has started stack. So, the stack will be loaded and after x seconds, the driver will start communicating, so that the stack will be running for sure at this point. This option can be usefull in combination with the Siemens S7 PLC-range. The communication card of these types of PLC tries to connect to the FactoryLink workstation whenever it encounters the physical address. As long as the stack is running, but FactoryLink NOT, incoming connection requests will be rejected. However the PLC tries to rebuild the connection after the rejection! This can cause an unwanted network load, by stopping the stack the network load is reduced.

Appendix C. SAPI Error codes

These error codes will be generated by the sapi dll and will be returned to the sapi_s7.exe
The Debug version of the driver will print the detailed error message in a dialog box with the type in the title box.

ERROR CODE	TYPE OF ERROR
S7_ERR_MAX_REQ	"connection handling"
S7_ERR_ORDERID_USED	"invalid order_id"
S7_ERR_NO_ERROR	"general errors"
S7_ERR_UNKNOWN_ERROR	“general errors"
S7_ERR_WRONG_CP_DESCR	"general errors"
S7_ERR_NO_RESOURCE	"general errors"
S7_ERR_INVALID_PARAMETER	"general errors"
S7_ERR_TOO_LONG_DATA	"general errors"
S7_ERR_TOO_MANY_DLL_USERS	"general errors"
S7_ERR_WRONG_IND_CNF	"invalid service"
S7_ERR_SERVICE_NOT_SUPPORTED	"invalid service"
S7_ERR_INVALID_CREF	"invalid communication reference (cref) or connection name"
S7_ERR_CONN_NAME_NOT_FOUND	"invalid communication reference (cref) or connection name"
S7_ERR_NO_LDB_XDB_FILE	"invalid communication reference (cref) or connection name"
S7_ERR_INVALID_ORDERID	"invalid order_id"
S7_ERR_INVALID_R_ID	"invalid r_id"
S7_ERR_R_ID_USED	"invalid r_id"
S7_ERR_NO_R_ID	"invalid r_id"
S7_ERR_OBJ_UNDEFINED	"invalid object"
S7_ERR_OBJ_ATTR_INCONSISTENT	"invalid object"
S7_ERR_OBJ_ACCESS_DENIED	"invalid object"
S7_ERR_INVALID_DATA_SIZE	"invalid data transfer"
S7_ERR_RECEIVE_BUFFER_FULL	"invalid data transfer"
S7_ERR_INVALID_DATARANGE_OR_TYPE	“invalid data transfer"
S7_ERR_INVALID_SEGMENT	"invalid data transfer"
S7_ERR_FW_ERROR	"controller or CMI or FW problems"
S7_ERR_MINI_DB_TYPE	"s7_mini_db_set() / s7_mini_db_get() errors"
S7_ERR_MINI_DB_VALUE	"s7_mini_db_set() / s7_mini_db_get() errors"
S7_ERR_SERVICE_VFD_ALREADY_USED	"multi user errors"
S7_ERR_SERVICE_CONN_ALREADY_USED	"multi user errors"
S7_ERR_CONN_ABORTED	"connection handling"
S7_ERR_INVALID_CYCL_READ_STATE	"state of cyclic read"
S7_ERR_INSTALL	"SINEC problems"
S7_ERR_INTERNAL_ERROR	"SINEC problems"
S7_ERR_NO_SIN_SERV	"SINEC problems"
S7_ERR_NO_LICENCE	"SINEC problems"
S7_ERR_SYMB_ADDRESS	"symbolic address problems"
S7_ERR_SYMB_ADDRESS_INCONSISTENT	"symbolic address problems"
S7_ERR_REM_BRCV	"remote block problems”
S7_ERR_REM_BSEND	"remote block problems"
S7_ERR_REM_BSEND_CANCEL	"remote block problems"
S7_ERR_REM_DATABASE_TOO_SMALL	"remote block problems"
S7_ERR_NO_RCV_BLOCK	"pmc problems"

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 DeltaLink Sapi S7 protocol driver in the run-time manager. The errors can be generated from different parts of the driver which will be listed here.

Error #	H1 data package errors
201	Unidentified data received.
202	Encode value not supported.
203	Unknown encode type.
204	The size of the H1 packet was too small
205	Error in received H1 protocol header

Error #	H1 data package errors
300	No error.
301	Incorrect Q/Z type at handling block.
302	Area not present in AG.
303	Area in AG too small.
304	AKD Error in the AG.
305	Error with condition code word.
306	No valid ORG format.
307	No free data buffer.
308	No free transport links.
309	Remote error with Read/Write.
310	Data Link error.
311	Handshake error.
312	Initiation error.
313	Abort after reset.
314	Job with boot strap function.
315	Job not present.
316	Area in AG too small.

Error #	FactoryLink errors
401	Internal error
402	Out of memory
403	Operating system error
404	Initialization not successful
405	Initialization not successful
406	Incorrect function
407	Incorrect argument
408	Incorrect data
409	Bad tag
410	Null pointer assignment
411	Change flag not set
412	Procedure table full
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	FL already active
423	No lock
424	Lock failed
425	Lock expired
426	Wait failed
427	Termination flag set
428	Q-size to big
429	Q-size changed
430	No tag list
431	Tag list changed
432	Wake up failed
433	No signals
434	Signaled
435	Not a mailbox
436	No messages
437	Access denied
438	Attribute failure
439	Invalid attribute
440	Attribute not defined
441	Application exists
442	RTDB does not exist
443	No task bit
444	Not a lite task

Error #	IMX errors
450	Bad message type
451	Message with dataset control tag not found in queue
452	No messages available to query
453	Bad receive mailbox tag
454	Bad mailbox send tag
455	Bad dataset control tag
456	Message cannot be adjusted
457	Operation to big for variable
458	Unknown boundary
459	Function not supported
460	No message for this index present
461	The remote dataset Is not defined on this system
462	The received dataset was not registered
463	The message is not queued
464	Message is rejected due error in the remote IMX
465	Illegal method of addressing bits on bit boundary
466	Element cannot be written
467	Invalid buffer specified
468	Block write function impossible
469	Maximum number of messages in MBX reached
470	No memory left
471	Error registering standard dataset
472	Error writing message in pipe
473	Not supported IMX message
474	Error creating pipe
475	Error starting threat
476	Error server connecting to pipe
477	Error child connecting to pipe
478	Error reading the pipe
479	Error number of bytes read from pipe
480	Error writing into the pipe
481	Error number of bytes written into the pipe
482	Error reading dataset from the mailbox
483	No communication buffers assigned
484	Error decrementing semaphore
485	Error writing to pipe, no space left
486	Error querying no. of messages for Ioxlator
487	Max. No. of messages in Ioxlator MBX reached

Appendix D. 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 "SAPI_S7". The error messages that may be displayed are as follows

Running in DEMO mode

The task did not found the DeltaLink protection code, when starting up. The option file could not be present or damaged. This is a non fatal error, however the task continues running in DEMO mode. The DEMO has a limitation timed for 5 hours. After shutdown the task can not be restarted without stopping the complete FactoryLink application.

Demo shutdown, licensed to run 5 hours

The task did not found the DeltaLink protection code, when starting up and continued running in DEMO mode. The DEMO has a limitation timed for 5 hours. After shutdown the task can not be restarted without stopping the complete FactoryLink application.

Demo restart prohibited, restart FactoryLink

The task did not found the DeltaLink protection code, when starting up. The option file could not be present or damaged. This is a non fatal error, however the task continues running in DEMO mode. The DEMO has a limitation timed for 5 hours. After shutdown the task can not be restarted without stopping the complete FactoryLink application.

Out of RAM

There is not enough RAM memory to load the complete configuration and/or task. This is a fatal error.

Error (%d) reading RTDB element

An error occurred reading a tag value. The error number is displayed.

Error (%d) forcing RTDB element

An error occurred forcing a tag value. The error number is displayed.

Error (%d) writing RTDB element

An error occurred writing a tag value. The error number is displayed.

Error (%d) retrieving event

An error occurred while waiting for an event to happen. The error number is displayed.

Can't open CT file

The task was unable to open the configuration table file, generally because it does not exist. This is a fatal error.

No triggers defined

The task was able to open the configuration table file, but contained not enough information to continue running. This is a fatal error.

Error reading CT index

An error occurred reading the index of the CT file. This is a fatal error.

Error reading CT header

An error occurred reading the header of a table in the CT file. This is a fatal error.

Error reading CT record

An error occurred reading a record of a table in the CT file. This is a fatal error.

Initialization critical parameters failed

Use of critical sections will not be possible, but is still required for full functionality of the protocol driver. This is a fatal error.

Event not registered to FL kernel
Tag event is not registered to the FactoryLink kernel. This is a fatal error.

No Ioxlators defined
There are no Ioxlators (or mailbox) defined. This is a fatal error.

No datasets defined
No dataset is defined, the protocol driver has nothing to do. Therefore this considered to be a fatal error.

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

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

IMX dataset not unique %s
IMX error, dataset control tag is not unique, the name of the dataset control tag is specified. This is a fatal error, a dataset control tag can only be used once, to define a dataset!

IMX (%d) Initialization failed
IMX initialization failed, the error number is reported. This is a fatal error.

IMX (%d) dataset registration failed %s
Registration of the reported dataset failed, the cause can be determined by the error number. This is a fatal error.

IMX semaphore creation failed
Creation of semaphores, needed for IMX failed. This is a fatal error.

IMX failed to start thread
The protocol driver could not start a thread. This is a fatal error.

IMX maximum msg for Ioxlator %d reached
The reported Ioxlator failed to process the queued messages, the configured maximum length of the queue is reached. The protocol driver stops queuing until the number decreases.

IMX (%d) send error to mailbox %s
The protocol driver could not queue a message for the reported Ioxlator due to an error, the corresponding error number is reported.

IMX (%d) reply to mailbox %s
The protocol driver could not queue a message for the reported Ioxlator due to an error, the corresponding error number is reported.

Com building device failed
Communication error building of a communication device failed. This is a fatal error.

Com unknown DataSet of type %d
Unknown dataset type specified, the type invalid type id is reported. This is a fatal error.

Com init comm. layer %d
An error occurred during initialising the communication layer Specified is the error number.

Com (%d) connecting '%s' %s

An error occurred during connecting to a device. Specified are the device name, functionality and the error number.

Com (%d) listenening %s %s

An error occurred during listening. Specified is the error number.

Com (%d) accepting connection %s

An error occurred during accepting a connection to a device. Specified are the device name and the error number.

Com (%d) disconnecting '%s' %d

An error occurred during disconnecting from a device. Specified are the device name, functionality and the error number.

Com (%d) read %s

An error occurred during reading from a device. Specified are the device name and the error number.

Com (%d) reading for exception %s

An error occurred during reading from a device for an exception write. Specified are the device name and the error number.

Com (%d) encode write %s

An error occurred during writing for an encoded write to a device. Specified are the device name and the error number.

Com (%d) write %s

An error occurred during writing to a device. Specified are the device name and the error number.

Com (%d) receive %s

An error occurred during receiving from a device. Specified are the device name and the error number.

Com (%d) reject connection %s, %s

An incoming connection request is rejected. Specified is the error number.

E. Installation of CP1413 card, CP-PG interface

For every connection a different vfd is used, the read the write and the recv connection can use one vfd name as well but this is not currently tested. The TSAP contains two octets the first octet is 10 for the read, the second octet is 11 for the write, the third is 12 for the recv. The second octet contains the slot number of the S7-400 CPU.

In our example the CPU is positioned at the third slot.

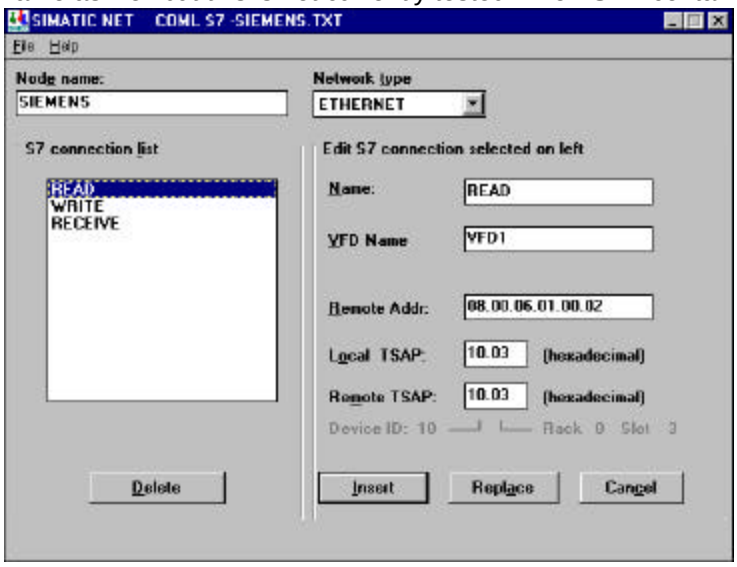


Figure E.1.1 COML_S7.

The Siemens CP-card can be tested, reset and restarted with the PG/PC Interface dialog box from COML_S7. After changing the Ethernet(MAC)-Address one can restart the cp-card.

The PG/PC Interface is used to select the CP board in the PC. When you click on Properties you will get the right dialog box with the tab Ethernet(MAC_Address), Mode and S7_Protocol.
With the ethernet mac-address you can select the database generated by the “COML S7” configuration tool from simatic net. In “COML S7” you can select the protocol out of profibus, industrial ethernet and tcp/ip. We choose one remote addres for the read, write and recv connection.

Figure E.1.2 Setting the PG/PC Interface.

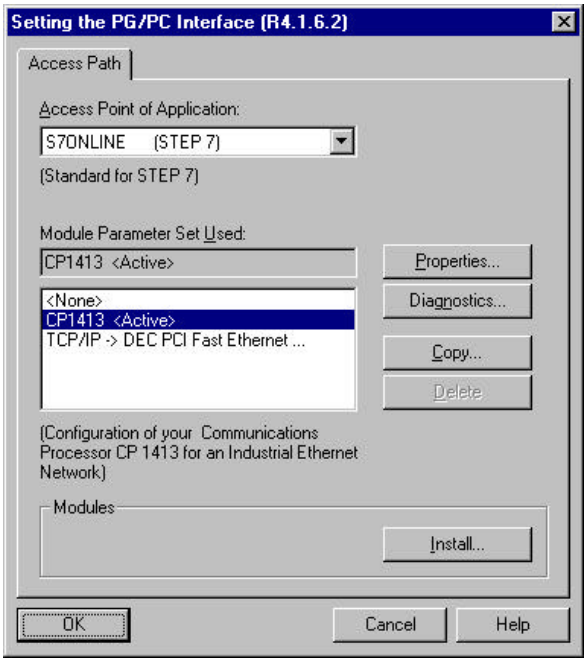
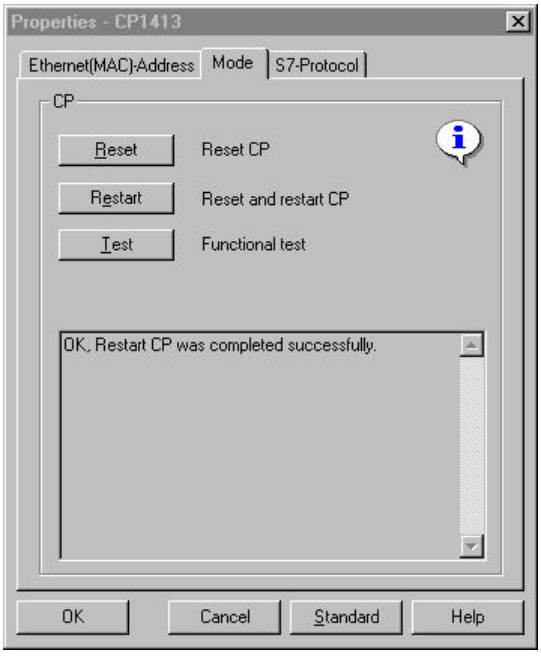


Figure E.1.3 Properties.



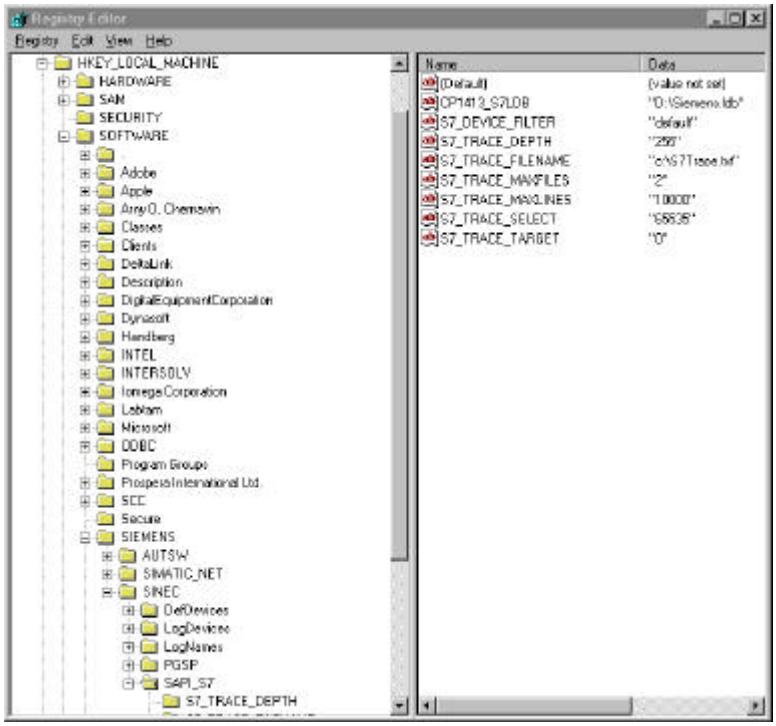
We generate the binary db with “Check and Save Binary Database” and the *.ldb file will be generated.

E.1 Configuration

All the siemens SIMATIC NET features can be used in combination with the SAPI_S7 driver. For detailed trace files one can change the registry values in the registry such as TRACE_DEPTH.

E.2 Configuring the TRACE DEPTH of SAPI

Figure E.1.4. Regedit with registry entries.



As mentioned the ldb is automaticaly inserted into the registry with the CP1413_S7LDB name. The S7_TRACE_FILENAME has to be entered manually. This trace-file “S7Trace.txt” is overwritten if the sapi driver is starting.

E.3 PLC Software example

```
DB100

   0     KC    =     'DB';        Source type
   1     KF    =     +00010;      Source DB number
   2     KF    =     +00001;      Source start address
   3     KF    =     +00005;      Source length
   4     KC    =     'DB';        Destination type
   5     KF    =     +00010;      Destination DB number
   6     KF    =     +00001;      Destination start addres
   7     KF    =     +00005;      Destination length
   8     KF    =     +00000;
   9     KF    =     +00000;
  10           KF    =     +00000;
```

Appendix F. Siemens S7-PLC

F.1 S7 H1 protocol

The Sapi S7 protocol driver is capable of communicating with a PLC from the Siemens S7 range. The S7 range of PLC's supports H1 communication with the following restriction No read/write header information is incorporated in the transmitted data packages. The consequence are the read functionality is no longer supported, for write and receive only one dataset (for each functionality) is useful for every device. When defining more then one for e.g. the write functionality means that all data is written to the same location in the PLC. When using this type of communication, the device type specified in the Sapi S7 protocol driver tables must be 'S7RW'!

A solution for this shortcoming is to use the S7 function blocks (FB15, FB16, FB17) which implements the SAPI blocksend and receive protocol on the PLC side. To enable the blockread functionality FB15 should be called once every PLC cycle, the same applies to the write functionality call FB16 every PLC cycle. The last function block FB17 can be used for the unsolicited receive. All the functions call internally the following functions

SFC20, SFC64	These are standard function blocks.
SFB12, SFB13	These function block are part of the SAPI package and implement the basic block receive and block send on the PLC side.

The functions blocks (FB15, FB16, FB17) for implementing the SAPI block send and receive on a S7 PLC, use the function blocks FC15, FC16, which are part of the SAPI package. These two functions implement the block read (FC15) and the block write (FC16) service on a connection. There is one important limitation for both function blocks the maximum length of the data package is 8192 bytes! From these 8192 bytes are 16 bytes needed to implement the H1 protocol, this result in an effective data length of **8176** bytes**!** When defining datasets for the Sapi S7 protocol driver this length of 8176 bytes is the maximum length of a dataset, even if the range inside the PLC can be larger.. Trying to write or read a dataset with a length exceeding the 8176 bytes will result in an error.

The supported data areas in the S7 PLC, including the maximum addresses and lengths are summarised in the table below.**Supporte Data areas Siemens S7**

Data area	Max. Db no.	Max. Address	Description
DB	4095	65534	Data block
FB		2047	Flag byte
IB		8191	Input byte
OB		8191	Output byte

FB15 S7_READ

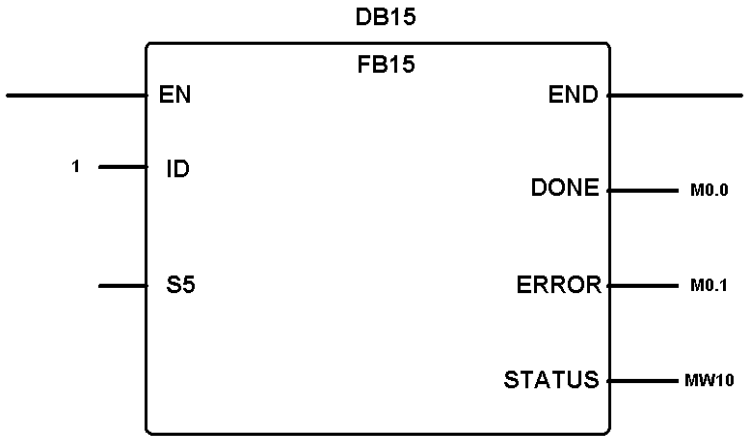


Figure F.1.1 Function block FB 15

This function block is used to implement the read functionality for the driver, when this function is used the PLC device type definition should be set to ‘S7’ . The choice between ‘S7’ depends on the desired addressing mode of data blocks. When using ‘S7’ data blocks are addressed with words, the ‘normal’ addressing mode for the S7 PLC range. Specifying ‘S7’ means that the driver confirms to the addressing mode on bytes for a data block in a S7 PLC. Note that this information also must be specified as a parameter for the function block. Left side of the function block is the input and the right side of the function block is output.

Parameters

Instance DB	The function block needs an instance DB, specify the desired block number.
ID	Id of the virtual connection which will be used, id can be found in the connection definition table.
S5	Boolean to specify if S5 or S7 addressing mode is to be used. A value of true means that the S5 addressing mode is to be used. S7 addressing mode is used when the value is FALSE.
DONE	Output flag set to TRUE for one cycle on completion of a read job, without an error.
ERROR	Output flag set to TRUE for one cycle on completion of a read job, with an error.
STATUS	Output word set to zero on completion of a read job, without an error. If a job ends with an error, the word contains an error code from the one of the following function blocks SFC20, SFB12 or SFB13
SFB12	BSEND
SFB13	BRCV

FB16 S7_WRITE

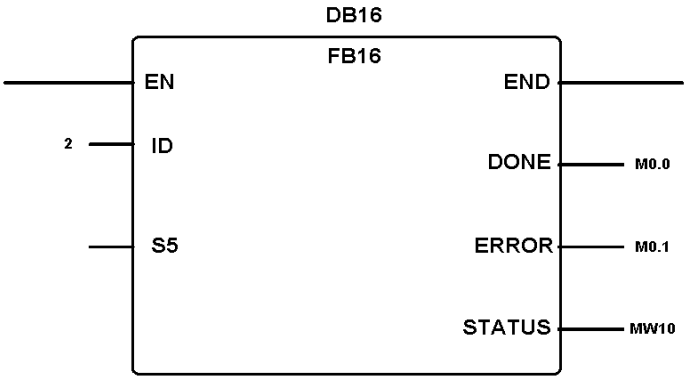


Figure F.1.1 Function block FB16

This function block is used to implement the write functionality for the driver, when this function is used the PLC device type definition should be set to ‘S7’ . The choice between ‘S7’ depends on the desired addressing mode of data blocks. When using ‘S7’ data blocks are addressed with words, the ‘normal’ addressing mode for the S7 PLC range. Specifying ‘S7’ means that the driver confirms to the addressing mode on bytes for a data block in a S7 PLC. Note that this information also must be specified as a parameter for the function block.

Parameters

Instance DB	The function block needs an instance DB, specify the desired block number.
ID	Id of the virtual connection which will be used, id can be found in the connection definition table.
DONE	Output flag set to TRUE for one cycle on completion of a write job, without an error.
ERROR	Output flag set to TRUE for one cycle on completion of a write job, with an error.
STATUS	Output word set to zero on completion of a write job, without an error. If a job ends with an error, the word contains an error code from the one of the following function blocks SFC20, SFB12 or SFB13
SFB12	BSEND
SFB13	BRCV

FB17 S7_SEND

This function block is used to implement the unsolicited receive functionality for the driver, when this function is used the PLC device type definition should be set to ‘S7’ . The choice between ‘S7’ depends on the desired addressing mode of data blocks. When using ‘S7’ data blocks are addressed with words, the ‘normal’ addressing mode for the S7 PLC range. Specifying ‘S7’ means that the driver confirms to the addressing mode on bytes for a data block in a S7 PLC. Note that this information also must be specified as a parameter for the function block. Additional parameters include a start condition and a definition of the data area which should be send to the FactoryLink workstation. If the start condition is false, no data transmission will be initiated, but the status of a pending job will be reported on the output parameters.

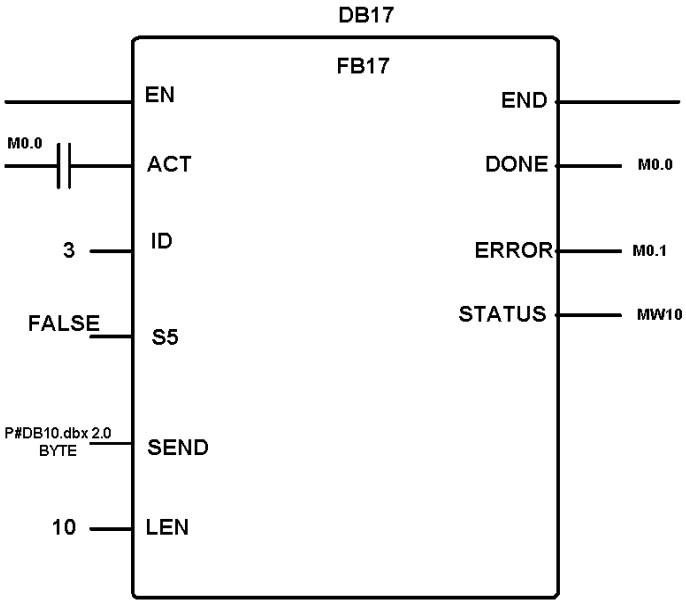


Figure F.1.2 Function block FB17

The left side of the function block is input and the right side of the block is output.

Parameters

Instance DB	The function block needs an instance DB, specify the desired block number.
ACT	Activate send command, in the value is TRUE a send command is started.
ID	Id of the virtual connection which will be used, id can be found in the connection definition table.
SEND	Any pointer to identify the area to send to the FactoryLink workstation. Specify the start address and the maximum length of this area. The actual length to send is specified with the next parameter. Note that the maximum length must be greater or equal to the maximum length.
LEN	Actual length in bytes to send.
DONE	Output flag set to TRUE for one cycle on completion of a write job, without an error.
ERROR	Output flag set to TRUE for one cycle on completion of a write job, with an error.
STATUS	Output word set to zero on completion of a write job, without an error. If a job ends with an error, the word contains an error code from the one of the following function blocks SFC20, SFC64, SFB12 or SFB13

F.2 Encoded write handling

The Sapi S7 protocol driver is capable to perform an encoded write. An encoded write causes the protocol driver to place coded information in a data block in the PLC instead of directly updating or changing a variable.

The information, placed in the PLC by an encoded write, should be decoded by a PLC-program in order to update/change the correct variable. A standard function block (FB10) is provided by DeltaLink for decode actions inside the PLC. The standard function block FB10 is called "ENCODED", and uses SFC20 and an instance data block. The instance data block should contain the information from an encoded write. The protocol driver places the information in a special way in the encode DB.

Encode DB		
Data byte	Type	Description
DBB0	KF	Reserved
DBB1	KF	Reserved
DBB2	KY	Destination
DBB3	KY	destination type
DBB4+5	KY	DB no..
DBB6+7	KY	address no.
DBB8	KY	Bit no.
DBB9	KY	Change flags
DBB10	KY	Value
DBB11	KY	Value
DBB12	KY	Value
DBB13	KY	Value

Figure F.2.1 Encode Data Block.

Destination the information in this byte indicates the destination of the value which should be updated.

Value		**Description**
M	=	Flag type.
D	=	Data block.
A	=	Output.
E	=	Input.

Destination type the information in this byte indicates the destination type of the value which should be updated.

Value		**Description**
F	=	Flag for DB, F, A or E.
B	=	Byte for DB, F, A or E.
W	=	Word for DB, F, A or E.
D	=	Double word, for long integer or float.

DB number The information in this byte is the number of a DB. If the destination is not D the value in this byte is irrelevant. The value is between 0 and 4095.

Address number address number of the byte, word, data word, timer or counter. The value is between 0 and 4095.

Bit number the number of the bit in a word or a flag. If the destination type is not flag, the value is irrelevant.

Change flags For every encoded write the protocol driver writes the value 255 in this byte. Function block FB10 uses this value to detect if there is new information.

Value value may be up to longs, depending on the type of data. For flag and byte values DBB16 is used. For word values DBW10 is used. And for long integers and float DBD8 plus DBW10 is used.

FB10 ENCODED

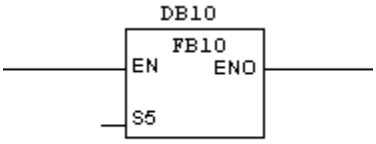


Figure F.2.2 Function Block FB10.

Parameters

Instance DB	The function block needs an instance DB, specify the desired block number.
S5	Boolean to specify if S5 addressing mode is to be used. A value of TRUE means that the S5 addressing mode is used. S7 addressing mode is used when the value is FALSE.

F.4 PLC Software example

```
ORGANIZATION_BLOCK OB 1
TITLE =S7 SAPI demo
VERSION  0.1


VAR_TEMP
  OB1_EV_CLASS  BYTE ;           //Bits 0-3 = 1 (Coming event), Bits 4-7 = 1 (Event
                                  //class 1)
  OB1_SCAN_1  BYTE ;             //1 (Cold restart scan 1 of OB 1), 3 (Scan 2-n of
                                  // OB 1)
  OB1_PRIORITY  BYTE ;           //1 (Priority of 1 is lowest)
  OB1_OB_NUMBR  BYTE ;           //1 (Organization block 1, OB1)
  OB1_RESERVED_1  BYTE ;         //Reserved for system
  OB1_RESERVED_2  BYTE ;         //Reserved for system
  OB1_PREV_CYCLE  INT ;          //Cycle time of previous OB1 scan (milliseconds)
  OB1_MIN_CYCLE  INT ;           //Minimum cycle time of OB1 (milliseconds)
  OB1_MAX_CYCLE  INT ;           //Maximum cycle time of OB1 (milliseconds)
  OB1_DATE_TIME  DATE_AND_TIME ; //Date and time OB1 started
END_VAR
BEGIN
NETWORK
TITLE =Handling of basic communication functions

      CALL FC     1 ;

NETWORK
TITLE =Timer/Counter for SAPI demo

      OPN   DB    50;
      L     T      1;
      T     DBW    2;
      T     MW     2;
      L     C      1;
      T     DBW    4;
      T     MW     4;
      AN    T      1;
      L     W#16#999;
      SD    T      1;
      A     M    100.2;
      CU    C      1;
      L     C      1;
      L     999;
      >=I   ;
      R     C      1;
END_ORGANIZATION_BLOCK
```

```
FUNCTION FC 1 VOID
TITLE =S7 H1
//The three functionality's for H1 communication with FactoryLink are handled
//here
//1. Read
//2. Write
//3. Unsolicited receive
NAME  H1
VERSION  0.1

BEGIN
NETWORK
TITLE =FactoryLink Read
//1. FactoryLink read addressing information is evaluated and the requested data
//   is send to the FactoryLink work station.
      CALL FB     15 , DB     15 (    // FactoryLink read, send requested data
           ID                        = 1,
           S5                        = FALSE,// S7 addressing mode
           DONE                      = M    100.2,
           ERROR                     = M    100.3,
           STATUS                    = MW   110

NETWORK
TITLE =FactoryLink Write
//2. FactoryLink write data including addressing info is received, addressing
//   info is evaluated, and data is set on the correct location.
      CALL FB     16 , DB     16 (    // FactoryLink write, move received data
           ID                        = 2,
           S5                        = FALSE, // S7 addressing mode
           DONE                      = M    100.4,
           ERROR                     = M    100.5,
           STATUS                    = MW   112

NETWORK
TITLE =FactoryLink unsolicited Receive
//3. FactoryLink unsolicited receive whenever there is received an encoded write,
//   the addressing information of this write is send back to the FactoryLink
//   workstation.
      L     DB10.DBB    17;          // Check if encoded write is received
      L     0;
      <>I   ;
      =     M    100.7;             // Encoded write received, send info back..
      CALL FB     17 , DB     17 (    // Send encoded write info to FactoryLink
           ACT                       = M    100.7,
           ID                        = 3,
           S5                        = FALSE, // S7 addressing mode
           SEND                      = P#DB10.DBX 2.0 BYTE 10,// Origin
           LEN                       = 10, // Length of data in bytes
           DONE                      = M    101.0,
           ERROR                     = M    101.1,
           STATUS                    = MW   114

      CALL FB    10 , DB    10 (    // Evaluation of encoded write
           S5                        = FALSE // S7 addressing mode

END_FUNCTION
```