PTS/Profibus Interface Software
User's Manual

Issue 2

April 1999

MAN537

Contents

1. Introduction

1.1 General

This manual relates to the following versions of software in the PTS unit:

Profibus Interface	Version 1.2
PTS Host software	Version 1.9.2 or later

This document describes the PTS/Profibus interface implemented using the COM-DPS interface of the CPU360 (PTS Mk2 or Machine Controller). The PTS acts as a Profibus-DP slave which is intended to be controlled by a master PLC or PC device on the Profibus network.

The Profibus-DP network is designed for high speed data communication between central controllers (PLCs, PCs, etc.) and distributed field devices (I/O, drives, etc.). At the physical level the network consists of a twisted pair, two wire cable using RS-485 signals at baud rates from 9.6 kB up to 12MB. Data communication between the central controller (the master) and the field devices (the slaves) happens cyclically. On each cycle the master device reads the input data from the slaves and writes the output data to the slaves. The amount of data read or written is limited to 512 bytes and is fixed during network configuration.

The master PLC/PC can access a user defined set of PTS variables for reading and writing. Up to 240 variables can be mapped onto Profibus. The variables can be used to control the PTS either as trigger variables to start actions or as parameters such as running speed, length, etc. The variables can also be used to pass operating information back to the PLC/PC such as positions, I/O status, etc. As well as variables the master PLC/PC can access the host I/O on the PTS. The host I/O consists of 64 virtual inputs and 64 virtual outputs which can be used for many of the same functions as the physical digital I/O. The PTS host I/O bits map onto the Profibus discrete I/O bits such that when the master node writes to an output bit the result appears in the corresponding PTS host input bit. Similarly when the master node reads an input bit it gets the contents of the corresponding PTS host output bit. Host inputs are supported by the DI, II and RI commands while host outputs are supported by the SO, CO, IO and RO commands.

1.2 Profibus Concepts

Unlike traditional asynchronous command-response protocols, Profibus-DP operates on data. Instead of providing a medium where network nodes can perform numerous instantaneous packet based conversations, Profibus-DP provides a medium where the network nodes all appear to share a block of memory which they can read from or write to at any time. In fact there are two such blocks involved known as Process Data Images because they provide an image of the process data on some network node. The two blocks are the Receive Process Data Image and the Send Process Data Image. The Receive Process Data Image receives its data from Profibus and is the image of the process data on the remote node. The Send Process Data Image sends its data to Profibus and is the image of the process data on the local node. The relationship between the send and receive process data images on the PLC and the PTS is shown in the following diagram.

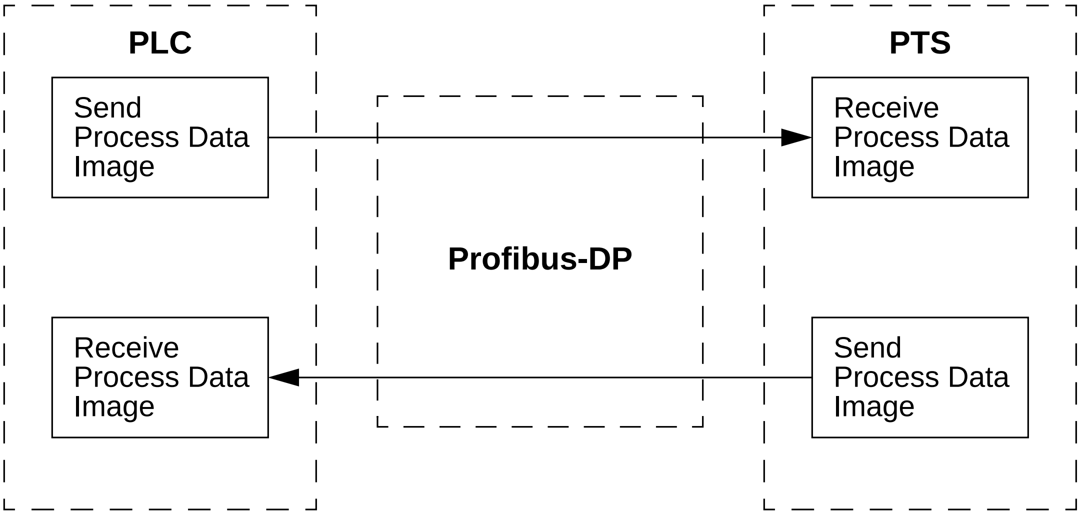

Figure 1. Process Data Images

1.3 Host I/O and Variable Mapping

In the PTS each process data image is a block of data up to 128 bytes long. The first 8 bytes (addresses 0 to 7) are always mapped onto the 8 host I/O groups. For the receive process data image the first byte at address 0 goes to host input group 10, the second byte at address 1 goes to host input group 11 and so on. For the send process data image the first byte at address 0 comes from host output group 10, the second byte at address 1 comes from host output group 11 and so on.

The remainder of each process data image starting at address 8 is mapped to a set of variables defined by the user. To make best use of the space available each variable can be mapped to a single byte, two bytes (a word) or four bytes (a long word or long). Mapping variables to a byte or a word increases the number of variables which can be mapped onto the process data image but restricts the range of the data which can be transferred. A byte can only hold values between 0 and 255. A word can only hold values between 0 and 65535. In contrast a long can hold values between ±2147483647.

The Profibus interface on the PTS works by scanning the receive and send process data images every 10 ms. approximately. When the receive image is scanned the first 8 bytes are sent to host input groups 10 to 17. If any variables are mapped the data is read from the receive image and if it has changed value since the last scan it is written to the appropriate variables. The reason for writing to the variables only when the data changes is to stop trigger variables being activated on each scan which would cause unnecessary load on the processor. When the send image is scanned the first 8 bytes are taken from host output groups 10 to 17. If any variables are mapped the data is read from the appropriate variables and placed in the send image. It is the responsibility of the underlying Profibus protocol to transfer the send image to the PLC and transfer the receive image from the PLC.

The following diagram shows an example of the receive process data image. The image has two variables mapped as bytes at addresses 8 and 9, two variables mapped as words at addresses 10 and 12 and one variable mapped as a long at address 14.

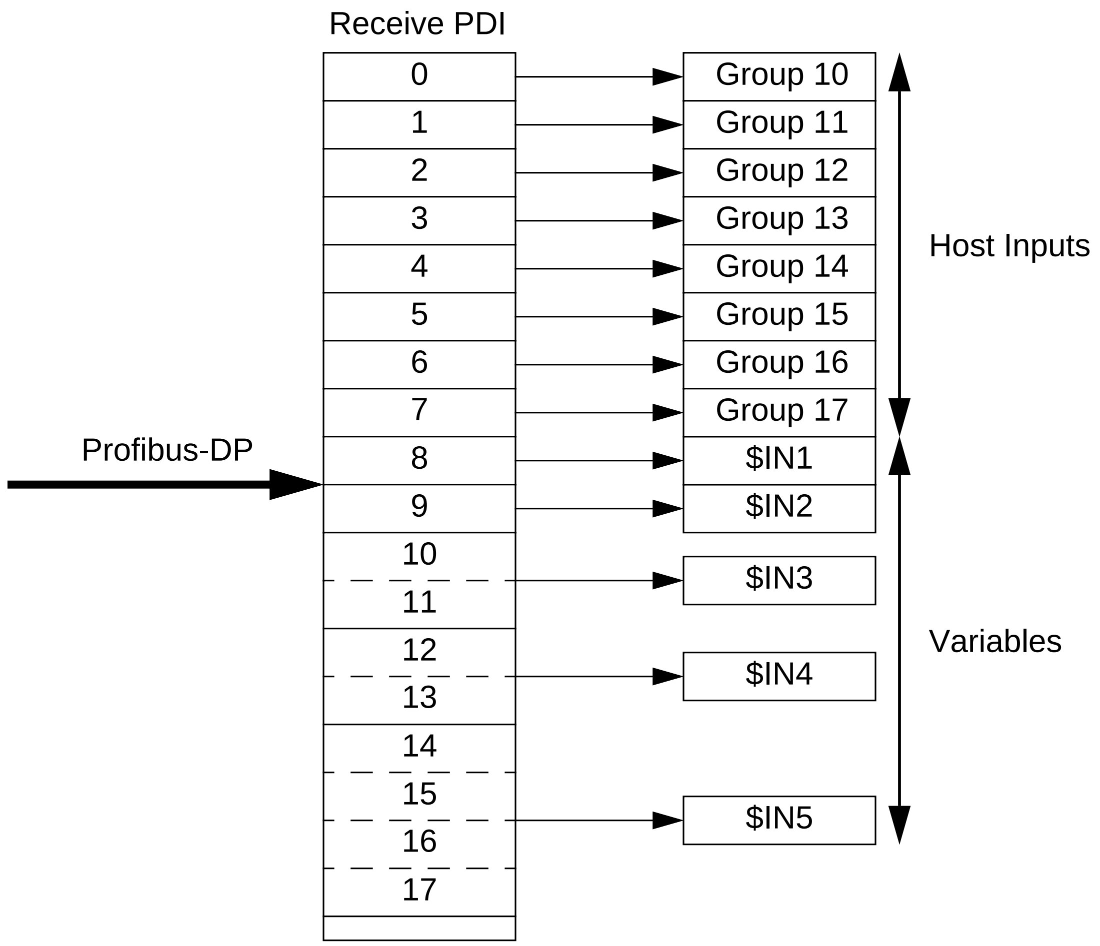

Figure 2. Receive Process Data Image Mapping Example

2. Configuring the PTS for Profibus

2.1 Hardware

Before switching the PTS on check that the COM-DPS interface board is in place on the CPU360 processor board. Without this interface board the Profibus software will not work. Also check that the CPU360 is an Issue E board or later.

2.2 Software License Key

The software for the Profibus interface will not operate unless a software key has been entered to enable this option. The software key is different for each PTS and can be obtained from your sales office given the system serial number which can be found by using the SK command as shown below.

To enable the software the following command should be entered on Port A (the main programming port) in privileged mode. You enter the text in **bold** while the PTS displays something similar to the rest.

```
1> SK
Serial number: 006545
Feature     Version   Key
New feature ? profibus
Version ? 1.2
Key ? abcd
OK
```

Note that the feature name (profibus) must be entered in lower case exactly as shown above. Note also that it is necessary to turn the power off and back on again to run the Profibus software. The key value is dependent on the unit serial number and should be obtained from your Quin sales office.

If the software needs to be disabled, first make a note of the software key in case it is needed in the future. Then proceed as above but simply press the Return key in response to the "Version ?" prompt as follows.

```
1> SK
Serial number: 006545
Feature     Version   Key
profibus     1.1     ABCD
New feature ? profibus
Version ?
Feature profibus removed
```


2.3 Configuration Shell

2.3.1 Accessing the Shell

The Profibus configuration shell is a command interface specific to Profibus which allows you to configure the variables for Profibus, save the configuration, etc.

To enter the shell type the PB command at Port A (the main programming port) in privileged mode:

```
1> PB

pb>
```

The pb> prompt shows that the configuration shell is ready for a new command. Typing **help** makes the shell display a list of available commands as follows:

```
pb> help
clear          Unmap all variables
default        Reset variable mapping to the default
force <var> [<addr>] [recv | send | both] [byte | word | long]
               Same as map but does not check for overlap
help           Display list of commands
list           Display current I/O and variable mapping
map <var> [<addr>] [recv | send | both] [byte | word | long]
               Map variable at optional specified address
quit           Quit from this shell
restore        Restore variable mapping from NVM
save           Save variable mapping to NVM
unmap <var> [recv | send | both]
               Unmap variable from receive, send or both maps
upload         Output mapping as commands to logfile
```

If Profibus is not enabled an error message is displayed instead:

```
1> PB

Profibus is not enabled

1>
```

In this case you need to enter a software license key as described in the section Software License Key and cycle the power to the PTS to start the Profibus software.

2.3.2 List Variables

The **list** command shows how host I/O and variables are mapped onto the receive and send process data images as shown in the following example:

```
pb> list
Address   Receive    Address   Send
  0       I/P Grp 10   0       O/P Grp 10
  1       I/P Grp 11   1       O/P Grp 11
  2       I/P Grp 12   2       O/P Grp 12
  3       I/P Grp 13   3       O/P Grp 13
  4       I/P Grp 14   4       O/P Grp 14
  5       I/P Grp 15   5       O/P Grp 15
  6       I/P Grp 16   6       O/P Grp 16
  7       I/P Grp 17   7       O/P Grp 17
  8        V1 (4)      8        V1 (4)
 12        V2 (4)     12        V2 (4)
 16        V3 (4)     16        V3 (4)
 20        V4 (4)     20        V4 (4)
 24        V5 (4)     24        V5 (4)
```

The left hand pair of columns show the mapping in the receive process data image (data received from the PLC by the PTS). The right hand pair of columns show the mapping in the send process data image (data sent to the PLC by the PTS). The address column shows the byte address within the process data image for the object in the next column. The first 8 addresses are reserved for host I/O groups 10 to 17. Addresses above 8 are used for variables. In this case the receive and send columns show the variable name followed by the number of bytes mapped in brackets.

2.3.3 Map a Variable

The **map** command maps the specified variable onto Profibus at the address given. If no address is given the variable is mapped at the first available address if possible.The variable can be mapped onto the receive, send or both process data images by specifying **recv**, **send** or **both** respectively. By default the variable is mapped onto both images.The number of bytes mapped can be 1, 2 or 4 by specifying **byte**, **word** or **long** respectively. By default the variable is mapped as 4 bytes. If a variable is mapped as less than 4 bytes its range is restricted as shown in the table below.

Parameter	Options	Default	Comments
Variable	None	None	Must be specified
Address	None	First available	
Image	recv	both	
	send		
	both		
Size	byte	long	Range 0 to 255
	word		Range 0 to 65535
	long		Range ±2147483647

Table 1: Map Command Parameters

The example shows variable $V10 being mapped at address 20 in the receive image only with a size of 2 bytes.

```
> map v10 20 recv word
```

When a variable is mapped at a specific address a check is made to see if there is already another variable overlapping the space. If there is, a message similar to the one below is displayed asking if you want to replace the current variable:

```
Mapped address overlaps a receive variable
Do you want to overwrite with new variable ? (Y/N)
```

The new mapping is only made if you respond with Y or y to the message. If the new variable overlaps in the receive and send images, two messages are displayed.

The next example shows variable $V11 being mapped at the next available address in both images with a size of 4 bytes.

```
> map v11
```


2.3.4 Unmap a Variable

The **unmap** command unmaps the specified variable. If **recv**, **send** or **both** is specified the variable is unmapped from the receive, send or both process data images respectively. By default the variable unmapped from both images. The following example shows variable $V10 being removed from the receive image only.

```
> unmap v10 recv
```

2.3.5 Force Variable Mapping

The **force** command does the same as the map command but does not ask for confirmation if you are about to overwrite an existing variable. It is not intended to be used directly for configuring the variable map as it is less safe than the map command. It is provided mainly for use by the upload command described below.

2.3.6 Map Default Variables

The **default** command maps the default set of variables onto Profibus. This results in variables $V1 to $V30 being mapped as 4 bytes on both process data images from address 8 to 124 respectively. Together with the host I/O mapped at address 0 to 7 this completely fills both images.

2.3.7 Clear Mapping

The **clear** command clears the variable mapping. It is equivalent to unmapping all mapped variables. It is normally used to clear the variable map to known empty state before downloading a variable configuration file.

2.3.8 Save Configuration

The **save** command saves the current variable mapping to non-volatile memory. This configuration will be restored automatically when the unit next powers up. The variable mapping is not automatically saved so it is important to use the save command whenever you change the variable mapping so that the changes are not lost when you power down.

2.3.9 Restore Configuration

The **restore** command restores the variable mapping saved in non-volatile memory. The configuration is automatically restored from non-volatile memory on power-up. This command is mainly used to restore the configuration to a known state after you have been experimenting with different settings.

2.3.10 Upload Configuration

The **upload** command causes all the mapped variables to be listed to the screen in the form of shell commands. If the result of upload is captured to a file it can be downloaded later to set the variable map to the same state.

2.3.11 Exit from Shell

The **exit** or **quit** command exits from the shell back to the normal PTS prompt after applying changes.

3. Programming the PTS

3.1 Host I/O

The PTS host I/O bits map onto the Profibus I/O bits such that when the master node writes to an output bit the result appears in the corresponding PTS host input bit. Similarly when the master node reads an input bit it gets the contents of the corresponding PTS host output bit. The PTS currently supports 8 host input groups (10 - 17) which map onto addresses 0 to 7 in the receive process data image and 8 host output groups (10 - 17) which map onto addresses 0 to 7 in the send process data image. Note that the host I/O is system wide and is not channel or node specific like the standard I/O.

The host inputs are supported by a subset of the normal input commands as follows:

- BIg:[n] Inhibit function input.
- DIg:n /... Define function input (restricted).
- EIg:[n] Enable function input.
- IIg:n If input true do command line.
- LIg List input line definitions.
- MIg:[n] Mask function input.
- RIg:[n] Read input line(s) in group g.

The host outputs are similarly supported by a subset of the normal output commands as follows:

- COg:[n] Clear output line n in group g.
- IOg:n If output true do command line.
- LOg List output line definitions.
- ROg:[n] Read output line state(s) in group g.
- SOg:[n] Set output line n in group g.

The following example shows host input line 10:5 being defined as a function input to execute sequence 200. When the Profibus master sets output number 5 then the corresponding host input is set and the function input is triggered to execute sequence 200.

```
DI10:5+/XS200
```


3.2 Variables

In the PTS/Profibus interface up to 120 variables can be mapped on to either process data image to be sent or received over Profibus. Note however that some PLCs may be restricted in the size of the process data image they can handle. The Siemens CPU315 PLC for example can only map up to 64 variables in each process data image.

The variable database is a centralized facility which is accessible to all tasks in the PTS system and holds a set of integer variables. Because variables are generally accessible, it is possible for the user to change a variable via Profibus and for the variable to be used subsequently to set a motor parameter in the PTS. Similarly a variable can be set to some motor parameter, such as the position, which can then be read over Profibus. A variable can also be set up to trigger execution of a command string on the PTS.

A variable can be set to a constant value using ‘=’ (equals). For example the following command sets the variable $SPD to a value of 5000.

```
1> $SPD=5000
```

A variable can be used in place of a numeric parameter in most commands. For example the following command sets the velocity to the value of the variable $SPD which is currently 5000. If the variable has not been assigned a value, then the “undefined variable” error message is displayed.

```
1> SV$SPD
```

Conversely it is possible to query a parameter and place the result in a variable. The following example updates variable $SPD with the current velocity value.

```
1> $SPD=SV
```

A variable can be defined as a trigger variable so that when it is updated a string of commands is executed. The following example defines $SPD as a trigger variable which causes the velocity to be set to the value of $SPD each time the variable is updated.

```
1> $SPD>CH1/SV$SPD
```


4. Configuring Profibus

4.1 The COM-DPS Daughter Board

The COM-DPS daughter board is configured using the COMPRO program supplied by Hilscher. This is used to set the Profibus bus address and to set the size of the process data area which is being shared with the Profibus master device.

To run the COMPRO program, first disconnect the Profibus cable from the PTS and connect the serial cable, Hilscher part number KAB-SRV, between the top right hand connector on the PTS and the COMM 2 port of your PC. Run the Cprun_pts.bat file to start the COMPRO program.

The COMPRO program, Cprun_pts.bat file and QUIN database mentioned below are available from Quin Systems Ltd. on request.

4.1.1 Initial Download

To configure the COM-DPS initially select the QUIN database from the database menu in COMPRO using the arrow, Enter and Escape keys. This database sets the Profibus bus address to 1 and sets the process data size to 32 words in and 32 words out. Select **online** from the toolbar. Then select **database...** followed by **download**. This will download the pre-configured database. Finally select **system...** followed by **coldstart** to restart the COM-DPS with the new parameters. Close the COMPRO program, disconnect the serial cable and reconnect the Profibus cable to the PTS.

4.1.2 Modifying the Configuration

To modify the configuration it is best to upload the existing database in the COM-DPS, make the required modifications and then download the modified database. The upload the existing database, run COMPRO as described above but do not select a database to load. Dismiss the database menu using the Escape key. Select **online** from the toolbar. Then select **database...** followed by **upload**. An overview of the available modules will be displayed. Press Enter to start the upload.

To modify the uploaded database, select **database** from the toolbar followed by **edit...**. To change the bus address select **setup** which will display a dialogue box where you can change the bus address as required. To change the process data area select **modules** which will display a dialogue box where you can change the module type (in word, out word, etc.) and the module length. Having modified the database, you can save it to disk by returning to the database menu and selecting **save...**.

Finally download the modified database to the COM-DPS by selecting **online** from the toolbar. Then select **database...** followed by **download**. Finally select **system...** followed by **coldstart** to restart the COM-DPS with the new parameters. Close the COMPRO program, disconnect the serial cable and reconnect the Profibus cable to the PTS.

4.2 The Siemens PLC

4.2.1 Equipment Required

This section describes the configuration of a Siemens CPU315-2 DP PLC using the Siemens software package Step 7 version 3.1. If you are using a different PLC or different software refer to the manufacturers handbook. The example Step 7 project described here uses the following PLC components:

- PS307 5A power supply.
- CPU315-2 DP processor.
- 2 off SM374 digital I/O simulators.

If you are using different hardware components you will have to modify the example project accordingly. The example project, pts2, is available from Quin Systems Ltd. on request.

To programme the PLC you will also need:

- IBM PC or equivalent with at least 16MB memory.
- Siemens MPI card to communicate between the PC and PLC.
- Siemens Step 7 software.

Programming can also be done using a special purpose Siemens programming unit if available.

4.2.2 Configuring the PC

The PC is configured as follows:

- Install the MPI card in a spare slot in the PC.
- Connect the MPI card to the X1 MPI connector on the CPU315 using the cable supplied with the card.
- Connect the X2 DP connector on the CPU315 to the PTS S3A connector (top left hand side) using a suitable Profibus cable.
- Install the Step 7 software with the relevant license key.

4.2.3 Initial Download

Before running Step 7 copy the pts2 directory into the Siemens\Step7\S7proj directory and copy the Hil_7501.gsd file into the Siemens\Step7\S7data\Gsd directory. Start the SIMATIC Manager and select **File** from the toolbar. Select **Open**, then **Project** and open project pts2.

You should now see the standard hierarchy as shown below.

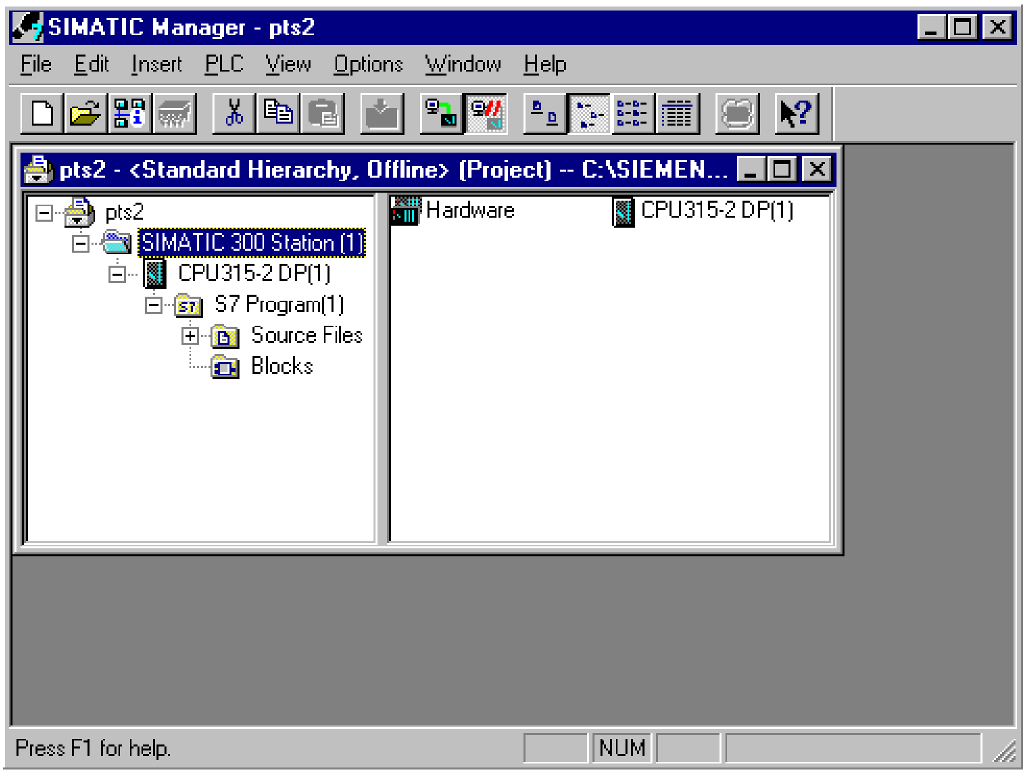

Figure 3. Example Project Standard Hierarchy.

This shows the project, pts2, which contains the SIMATIC 300 Station, i.e. the PLC rack, which in turn contains the CPU315 processor module. The CPU315 contains the S7 Program which in turn contains the Source Files and the Blocks. For the purposes of the example we will mainly be interested in the rack hardware and the program blocks.

Select SIMATIC 300 Station (1) in the left hand window then double click on **Hardware** in the right hand window. This should bring up the Hardware Configuration as shown below.

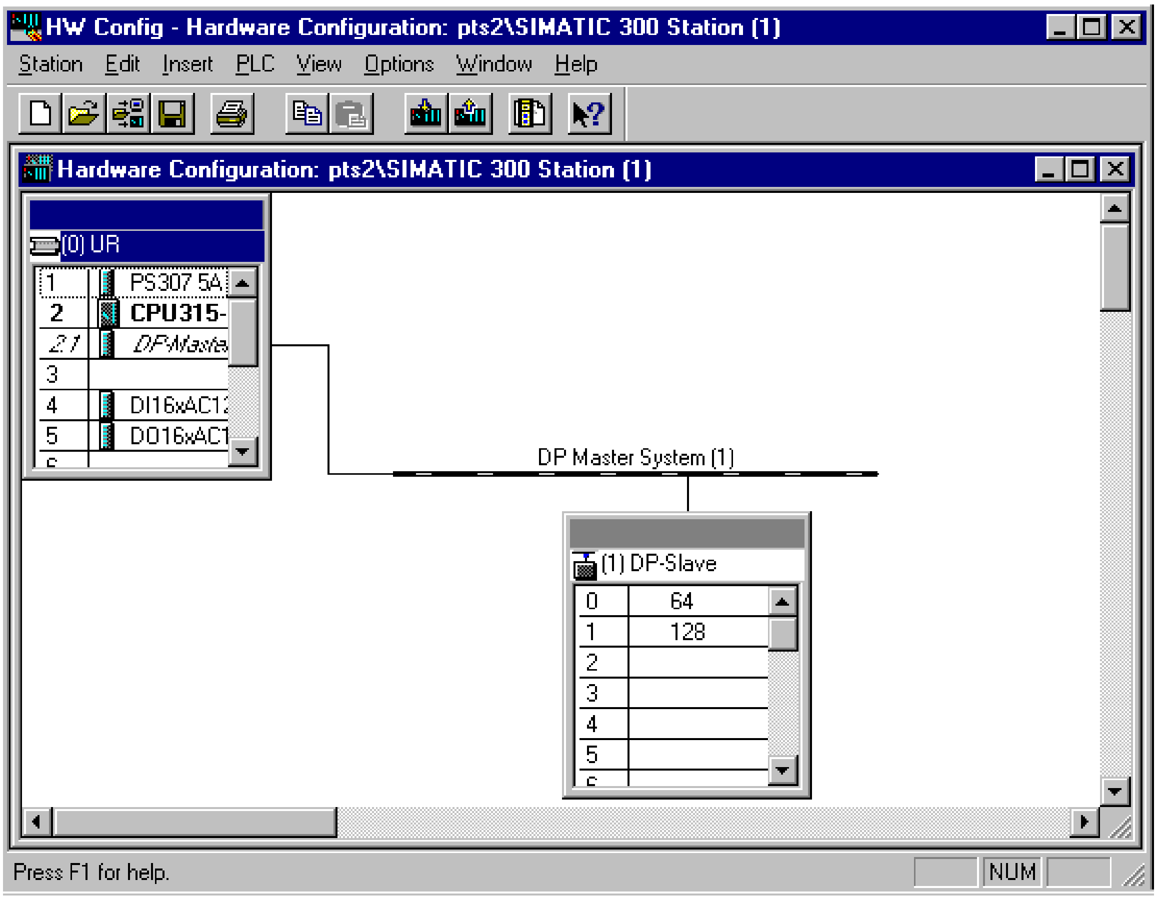

Figure 4. Example Project Hardware Configuration.

This shows the PLC rack on the left consisting of the PSU in slot 1, the CPU with Profibus DP master in slot 2 and the simulated digital inputs and outputs in slots 4 and 5 respectively.The Profibus DP slave, i.e. the PTS, is shown on the right with its input and output process data areas in slots 0 and 1.

To ensure that the COM-DPS daughter board is recognized select **Options** from the toolbar then select **Update DDB Files**. This checks all the available hardware description files into the hardware catalogue.

The next step is to download the hardware configuration to the PLC. Turn the keyswitch on the CPU module to **STOP**. Select **PLC** from the toolbar then double click on **Download**.

Once the hardware configuration is downloaded the next step is to download the PLC program. Return to the standard hierarchy window and select **Blocks** in the left hand window, then double click on **OB1** in the right hand window. This should bring up the ladder logic window as shown below.

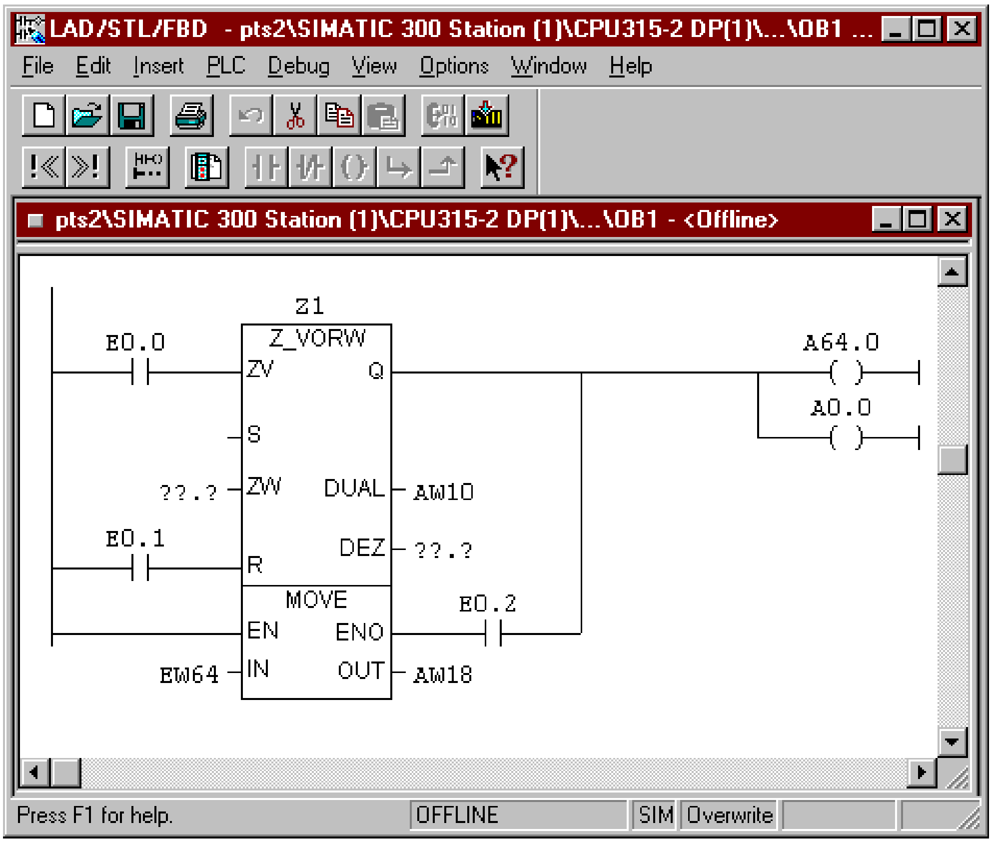

Figure 5. Example Project Ladder Logic.

Again select **PLC** from the toolbar followed by **Download** to download the program to the PLC. Once this is finished turn the CPU keyswitch to **RUN-P** to run the PLC. If everything is working correctly the SF DP and BUSF lights on the CPU module should be out.

4.2.4 Program Description

The example Step 7 project, pts2, consists of the hardware configuration for the PLC and Profibus network and the PLC program. The PLC can access 32 input words and 32 output words in the process data area starting at address 0. The PLC can also access 16 digital inputs and 16 digital outputs on the simulators starting at address 64. The ladder logic for the program is shown in the previous sub-section. The PTS should be set up to use the default variable mapping via the PB command which maps variables V1 to V30 as long words in both process data images at addresses 8 to 124 respectively. The host I/O groups are mapped on to the process data images at addresses 0 to 7.

The logic consists of an up counter, Z1 and a MOVE block. The up counter increments each time its ZV terminal sees a positive edge. The ZV terminal is controlled by input E0.0 which is mapped via Profibus to PTS host output 10:1. The value of the counter (between 0 and 999 only) is presented at the DUAL output which goes to output word AW10 which is mapped via

Profibus to the lower word of PTS variable V1. This whenever PTS host output 10:1 changes from 0 to 1 the counter increments and the value of V1 increments as well. The following fragment of PTS program shows how this can be demonstrated. The first two lines show the PTS commands to be entered. The last line is the result when the counter increments which causes variable V1 to change which triggers the LV action.

```
1: $V1>LV$V1
1: CO10:1/WT10/SO10:1
1: $V1=123
```

The counter can be reset to zero by a setting the R terminal to 1. This is controlled by input E0.1 which is mapped via Profibus to PTS host output 10:2. So the PTS command SO10:2 will reset the counter. The Q terminal of the counter is 0 if the counter value is zero or 1 otherwise. The Q terminal goes to output A64.0 which is a simulated output on the PLC and A0.0 which is mapped via Profibus to PTS host input 10:1. When the counter is reset the simulated output and host input 10:1 go to zero but when the counter starts to count up they go to one.

The MOVE block transfers the value of the IN terminals to the OUT terminals. In this case the simulated inputs at EW64 are transferred to AW18 which is mapped via Profibus to the lower word of PTS variable V3. When the simulated inputs are changed this is reflected in the value of V3.

4.2.5 Changing the Project

Changes to the example project can be made in two areas, namely the hardware configuration and the program logic. The Step 7 help facility can be used to find out how to make the required changes. The hardware configuration might be modified to change the Profibus addresses, to add modules to the PLC rack or to add Profibus slaves on the network. Whenever a change is made to the hardware configuration you must download the new configuration by selecting **PLC** then **Download** from the Hardware Configuration window. Changes to the program logic are made in the LAD/STL/FBD window. Whenever a change is made to the program logic you must download the new program by selecting **PLC** then **Download** from the LAD/STL/FBD window.

5. Hardware Configuration

5.1 Profibus Connections

The connections for the Profibus interface on the front panel 9 way socket S3A (top left hand connector) are shown below.

Pin no.	Signal	Pin no.	Signal
1	RGND	6	VP
2	N/C	7	N/C
3	RxD/TxD-P	8	RxD/TxD-N
4	N/C	9	N/C
5	DGND		

Table 2: S3A Profibus Connector

These connections are compatible with the standard Profibus cable supplied by Siemens.

5.2 Serial Connections

The connections for the COM-DPS daughter board serial programming port on the front panel 9 way plug S3B (top right hand connector) are shown below.

Pin no.	Signal	Pin no.	Signal
1	N/C	6	N/C
2	RXD	7	RTS
3	TXD	8	CTS
4	DTR	9	N/C
5	RGND2		

Table 3: S3B Serial Connector

These connections are compatible with the serial cable part number KAB-SRV supplied by Hilscher. Note that the serial cable should be disconnected while the Profibus cable is being used and vice versa.

5.3 LEDs

Underneath the Profibus connectors are a set of four green LEDs which show the state of the communications. The significance of the indicators is as follows:

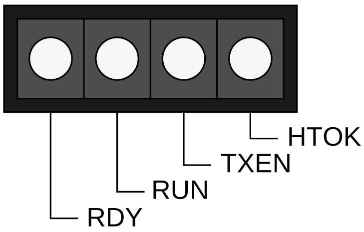

Figure 6. LEDs.

RDY — This LED is on when the daughter board is powered up.

RUN — This LED is on when the Profibus link is okay and the master and slave are communicating correctly. The LED flashes if the Profibus link is disconnected or the master and slave are not communicating, for example if their configurations do not agree.

TXEN — Transmit enable.

HTOK — This LED is on when the Profibus link is okay and the master and slave are communicating correctly. The LED is off if the Profibus link is disconnected or the master and slave are not communicating, for example if their configurations do not agree.

5.4 CPU360 Board Layout

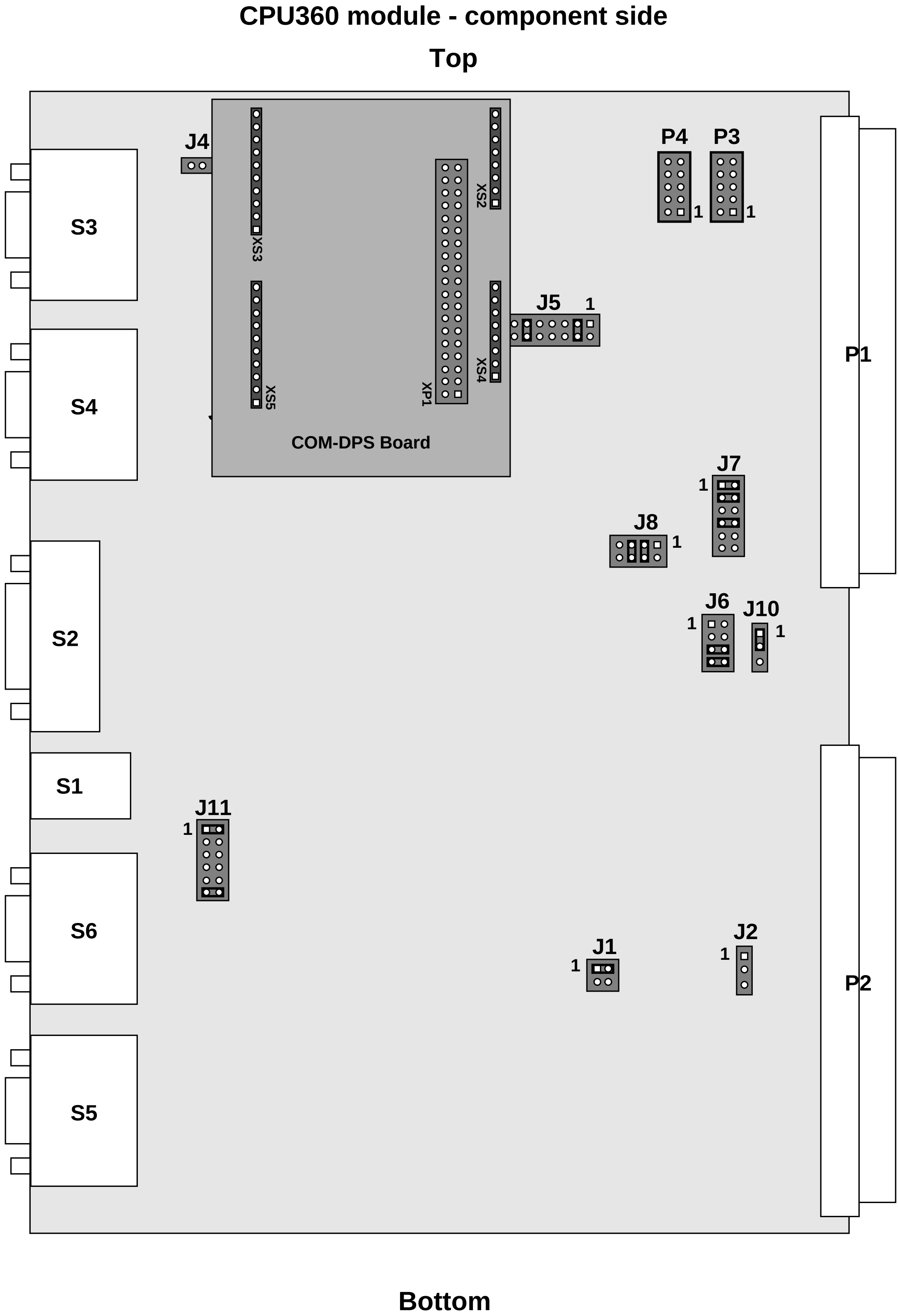

Figure 7. Jumper and connector locations

Index

O

P

Q

R

S

T

U

V

W