# sigmadue Control Unit CU-02

**User Manual**
**M.U. CPU-CB/CU-02-2/07.07**
**Cod. J30 - 478 - 1ACU02 E**



Copyright © 2005 ASCON spa

All rights reserved

No part of this document may be stored in a retrieval system, or transmitted in any form, electronic or mechanical, without prior written permission of ASCON SpA.

ASCON has used the best care and efforts in preparing this book and believes the information in this publication are accurate. The ASCON Products are subjected to continuous improvement, in the pursuit the technological leadership, these improvements could require changes to the information of this book. ASCON reserves the right to change such information without notice. ASCON makes no warranty of any kind, expressed or implied, with regard to the documentation contained in this book. ASCON shall not be liable in any event - technical and publishing error or omissions - for any incidental and consequential damages, in connection with, or arising out of the use of this book.

*sigma**due***®, *gamma**due***® *and delta**due***® are trademarks of ASCON spa.

All the other tradenames or product name are trademark or registered trademarks.

**ASCON spa**
Via Falzarego 9/11
20021 Baranzate (Milano) Italy
Phone ++39-02-333371 - Fax ++39-02-3504243
http://www.ascon.it
info@ascon.it

# *INDEX*


***Prerequisites*** . . . . . ***v***

Current Documentation on the Internet . . . . . v

***Chapter 1***

***Hardware description*** . . . . . ***1***

1-1 Architecture . . . . . 2

1-1-1 Communication ports . . . . . 2

1-1-2 Auxiliary digital I/O . . . . . 3

1-1-3 Diagnostics LEDs . . . . . 3

***Chapter 2***

***Installation*** . . . . . ***4***

2-1 Mechanical installation . . . . . 4

2-1-1 Installing modules and Removing modules . . . . . 4

2-2 Electrical installation . . . . . 4

2-2-1 Connect the communication cables . . . . . 4

2-2-2 Connector “A” connections . . . . . 5

***Chapter 3***

***CPU setup*** . . . . . ***6***

3-1 Connection of the CPU Setup Terminal . . . . . 6

3-1-1 RS232 Serial Communications Connection . . . . . 6

3-1-2 USB Serial Communications Connection . . . . . 7

3-1-3 Setting the Communications Parameters of the CPU RS232 Service Port . . . . . 7

3-1-4 Entering the Setup Menu . . . . . 8

3-2 Main Menu . . . . . 9

3-2-1 Network Setup Menu . . . . . 9

3-2-2 CPU Setup Menu . . . . . 9

3-2-3 CAN Setup Menu . . . . . 10

3-2-4 CPU Info Menu . . . . . 11

3-2-5 ETHERNET Setup Menu . . . . . 11

3-2-6 Serial Setup Menu . . . . . 12

3-2-7 CPU Setup Menu . . . . . 13

3-2-8 STARTUP Setup Menu . . . . . 13

3-2-9 Persistency Setup Menu . . . . . 13

3-2-10 CLOCK Setup Menu . . . . . 14

3-2-11 Modbus TCP/IP Setup . . . . . 14

3-2-12 Modbus TC/IP Secure ADDRS Table Menu . . . . . 15

3-2-13 Modbus TC/IP Priority ADDRS Table Menu . . . . . 15

***Chapter 4***

***Programming the CPU . . . . . 17***

4-1 Installing OpenPCS . . . . . 17
4-1-1 Hardware and Software Requirements . . . . . 17
4-1-2 Installation . . . . . 17
4-1-3 Starting OpenPCS . . . . . 17
4-1-4 Configuring OpenPCS . . . . . 18
4-2 OpenPCS Setup . . . . . 18
4-3 Software Utilities . . . . . 20
4-3-1 OPC Server . . . . . 20
4-4 Communication Ports Protocols . . . . . 20
4-5 Auxiliary digital port . . . . . 21
4-6 Watchdog Timer . . . . . 21
4-7 CANopen Extension for Ascon sigmadue CPU . . . . . 21
4-7-1 Introduction . . . . . 21

***Chapter 5***

***Ascon Function Blocks Libraries . . . . . 23***

5-1 AsconACLib . . . . . 23
5-2 AsconBasicIOLib . . . . . 24
5-3 AsconControlLib . . . . . 25
5-4 AsconCPULib . . . . . 25
5-5 AsconMBCommLib . . . . . 25
5-6 AsconLibrary . . . . . 26
5-7 Firmware Function Blocks List . . . . . 27

***Chapter 6***

***Technical data . . . . . 29***

6-1 General conditions . . . . . 29
6-2 Ambient conditions . . . . . 29

***Chapter 7***

***Reference documents . . . . . 30***

# *Prerequisites*

The products described in this manual should be installed, operated and maintained only by qualified application programmers and software engineers who are almost familiar with EN 61131-3 concepts of PLC programming, automation safety topics and applicable national standards.

## Current Documentation on the Internet

Make sure you are always working with the latest documentation published.

ASCON spa reserves the right to make any technical extension and change to the equipments that would serve the purpose of technical progress. Up to the time that a new manual revision is published, any update or change will be documented on the Internet at:

http://www.ascon.it

# *Chapter 1*
# *Hardware description*

.The system described in this User Manual is mainly composed of three components:

- Ascon sigma**due** series CPU (CU-02), ready to work with OpenPCS EN 61131-3 compliant programming system;
- Ascon sigma**due** I/O-CB remote I/O series of modules;
- Infoteam OpenPCS programming system.

sigma**due** CPU is a powerful processing unit, based on an ARM RISC processor, different types of memory and several communication ports.

sigma**due** I/O-CB is a complete family of I/O analogue and digital remote modules with special functions, based on the CANopen protocol over a CAN bus.

Infoteam OpenPCS is a powerful and useful standard programming system for PLC applications.

It is a clearly structured, easily operated tool for editing, compiling, debugging, managing and printing PLC applications in all the development phases.

OpenPCS supports EN61131-3 programming under Windows 2000, XP and Windows Server 2003.

Ascon sigma**due** line is based on the sigma**due** remote I/O system, combining its functionality with the capabilities of a PLC. The modular concept and the possibility of fieldbus networking means that you can adapt the system to your requirements quickly and easily. This gives the sigma**due** automation system an especially economical price/performance ratio.

This User Manual handbook introduces you to work with sigma**due** CPU line and the Infoteam OpenPCS programming system.

It explains how to install the hardware and software, and how to start up the system. Information on maintenance, troubleshooting and service are also included.

# 1-1 Architecture

From the programmer's point of view, a complete system is made up as in "*Figure 1.1 - Programming the sigmadue Control Unit"* below:

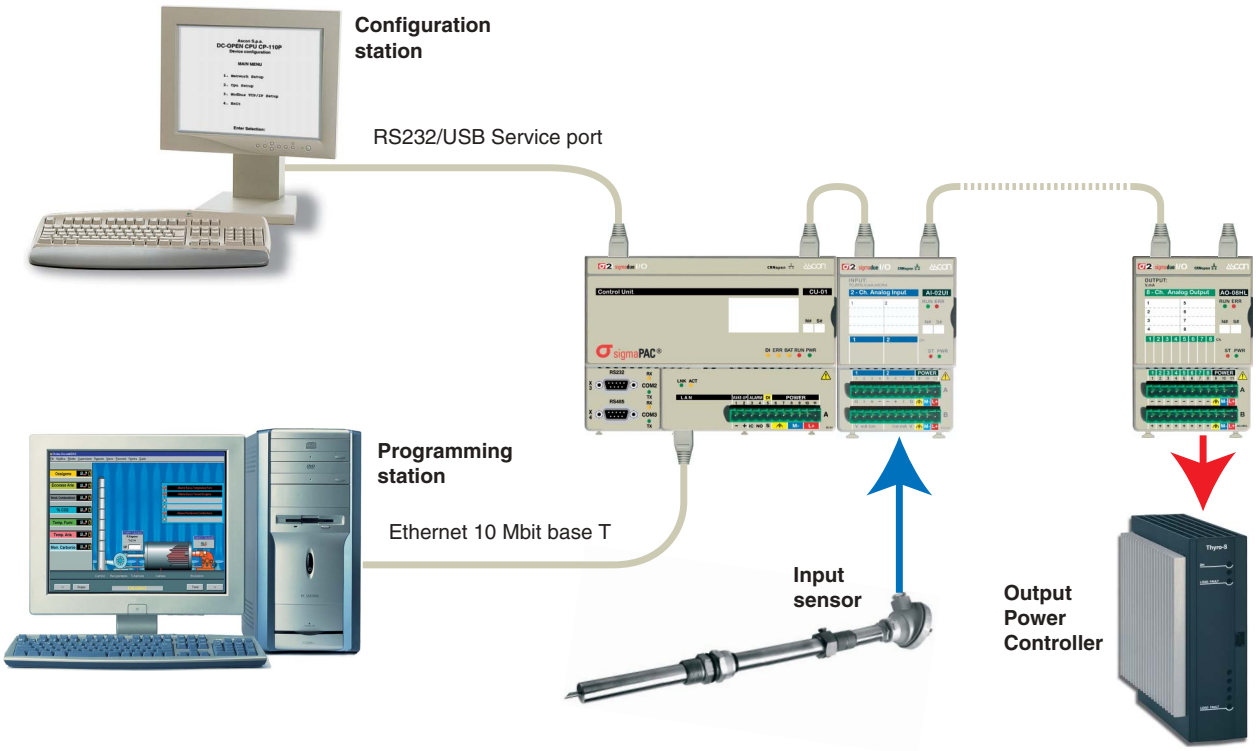



*Figure 1.1 - Programming the* sigma**due** *Control Unit*

In "*Figure 1.1 - Programming the sigmadue Control Unit"* the VT100 terminal and the PC with OpenPCS are displayed as two different devices, but the user may want to use just one PC to run both OpenPCS and a VT100 emulator (e.g. Hyper Terminal).

## 1-1-1 Communication ports

The CPU has 3 fixed communication ports (*see figure 2*):

- The CAN port will be used for the connection, through a CANopen network, to the distributed I/O;
- The Ethernet port (TCP/IP) will be used for the connection, through a LAN network, to the PC with the Infoteam OpenPCS environment, for:
  - Programming, Debug and commissioning;
  - Modbus TCP data exchange;
- The Service RS232 port will be used to:
  - Connect a VT100 terminal to perform the configuration of the device;
  - Standard ASCII serial port;
  - Modbus RTU data exchange.

In addition to the three mentioned fixed ports, a plug in board can be added (see Ordering Codes), by which the communication capabilities are increased, e.g.:

- Two channels RS232/485 with Modbus RTU protocol (master or slave);
- Profibus DP (slave);
- CAN – CANopen (slave).

Pinout of all communication Ports are described hereafter and in the "*CU02 Installation Manual*" **[6]**.

## 1-1-2 Auxiliary digital I/O

**DI** General Purpose Digital Input or RUN/STOP program functionality

**ALARM** General Fault Relay Output (IEC 61131 Watch Dog function);

**WAKE UP** Isolated General Purpose Digital Output (can be used, in conjunction with the Real Time Clock, for unattended applications).

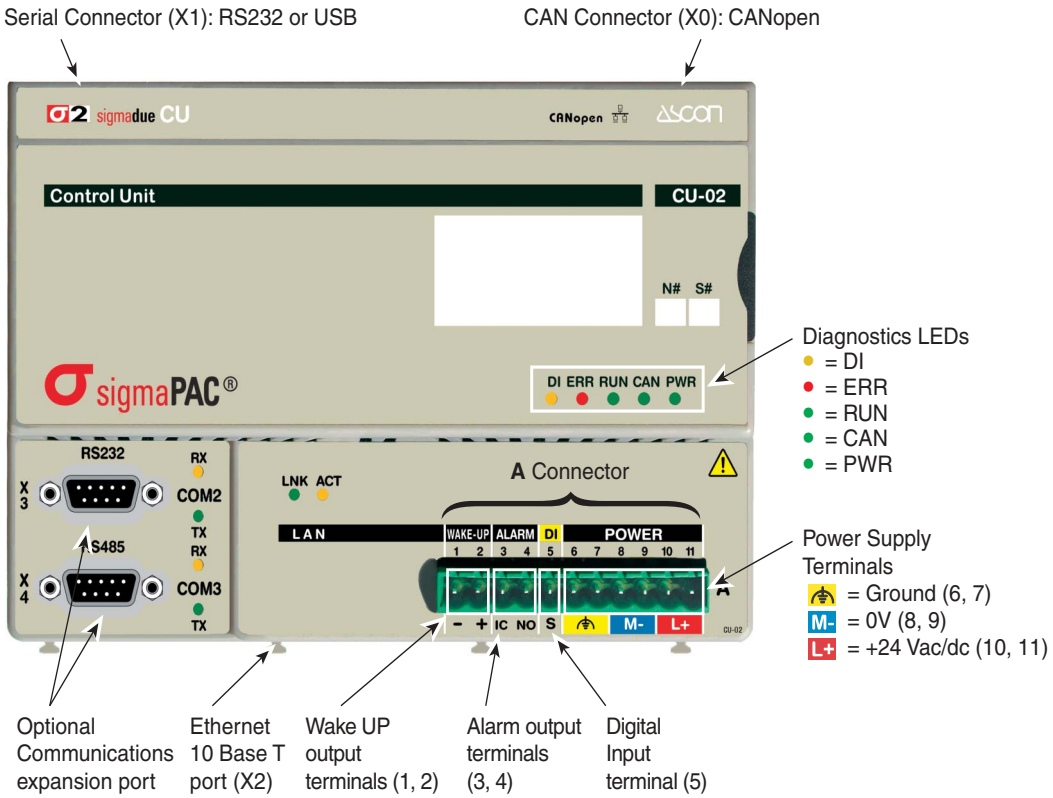



*Figure 1.2 - Control Unit I/O and Communication Ports*

## 1-1-3 Diagnostics LEDs

Referring to "*Figure 1.2 - Control Unit I/O and Communication Ports*" a description of the LEDs functions is given in the table below.

| LED | Color | Action | Description |
|---|---|---|---|
| DI | Yellow | ON | Digital Input active |
| ERR | Red | ON | Error on CANopen network |
| RUN | Green | ON | Program running |
| | | OFF | Program stop |
| | | Flickering | Back-Up battery low |
| | | Single flash | Configuration error, default set |
| | | Blinking | Retain varoiables restore error |
| CAN | Green | ON/OFF | RUN LED for CANopen Network |
| PWR | Green | ON | Power Supply ON |

*Table 1.1 - Diagnostics LEDs description*

# *Chapter 2*
# *Installation*

## 2-1 Mechanical installation

The sigma**due** Control Unit and the I/O modules are designed to be installed on standard DIN rails.
As the CPU has, on board, a CAN termination, it must be installed at one end of the CAN chain.
Up to 127 I/O modules can be connected in chain to each CPU. This value is the theoretical limit, Ascon spa reccomends to never exceed the number of 32 units.

### 2-1-1 Installing modules and Removing modules

A complete description on how the modules can be mounted on or removed from a DIN Rail can be found in the "*Control Unit CU-02 Installation Manual*" **[6]**.

## 2-2 Electrical installation

Refer to: "*Figure 1.2 -Control Unit I/O and Communication Ports*" "*Control Unit CU-02 Installation Manual*" **[6]** for details.

### 2-2-1 Connect the communication cables

*CANopen remote I/O Port*

For CANopen I/O modules connection (**X0**). I/O modules are connected with the included cables in a daisy chain fashion. The RJ45 type connectors have the pinout:

| **Pin** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
|---|---|---|---|---|---|---|---|---|
| **Signal** | CANH | CANL | CAN-GND | NC | NC | CAN-SHLD | CAN-GND | CAN-V+ |

*Serial Port*

**RS232 service port (X1)**
The connector X1 on the CPU module is an RJ45 type, with the following pinout:

| **Pin** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
|---|---|---|---|---|---|---|---|---|
| **Signal** | NC | NC | NC | GND | RX | TX | NC | NC |

**USB service port (X1)**
When installed , the connector X1 on the Contro Unit is a B type USB standard connector. The pinout of this cable is meaningless as the connection is standard.

*Ethernet 10baseT*

For OpenPCS development station (**X2**). The connector on the CPU module is an RJ45 type, with the following pinout:

| **Pin** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
|---|---|---|---|---|---|---|---|---|
| **Signal** | TX+ | TX- | RX+ | NC | NC | RX- | NC | NC |

### 2-2-2 Connector "A" connections

*Power supply*

Use the 6 poles on the right of the "**A**" connector and respect the polarity. Each of these terminals is doubled in order to allow the user to power, using an additional terminal block, other devices or sensors.

| **Pin** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Name** | - | + | IC | NO | S | FE | FE | M- | M- | L+ | L+ |
| **Function** | WAKE UP | | ALARM | | DI | F. EARTH | | POWER SUPPLY | | | |
| **Signal** | | | | | | FE | FE | **0V** | **0V** | **+24V** | **+24V** |

Power Supply

*Auxiliary ports*

The 5 poles on the left of the "**A**" connector are auxiliary ports.

| **Pin** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Name** | - | + | IC | NO | S | FE | FE | M- | M- | L+ | L+ |
| **Function** | WAKE UP | | ALARM | | DI | F. EARTH | | POWER SUPPLY | | | |
| **Signal** | COM | OUT | COM | OUT | INPUT | FE | FE | **0V** | **0V** | **+24V** | **+24V** |

Auxiliary ports

**Wake up** Software activated Digital Output. 24Vdc, 0.2A high side power switch, terminals 1 and 2;

**Alarm** Relay type digital output. SPST NO 24V, 1A, terminals 3 and 4;

**DI** 24Vdc digital Input, terminal 6. The return path can be linked to terminal M- (terminal 8 or 9) or to Wake-Up terminal (teminal 1).

# Chapter 3
# CPU setup

At start-up, the system starts a configuration session to perform the setup of the CPU devices. The user can insert the CPU setup data using a dumb VT100 terminal or a Personal Computer with Hyper Terminal program.

## 3-1 Connection of the CPU Setup Terminal

Depending on the CPU model, the X1 connector can be:

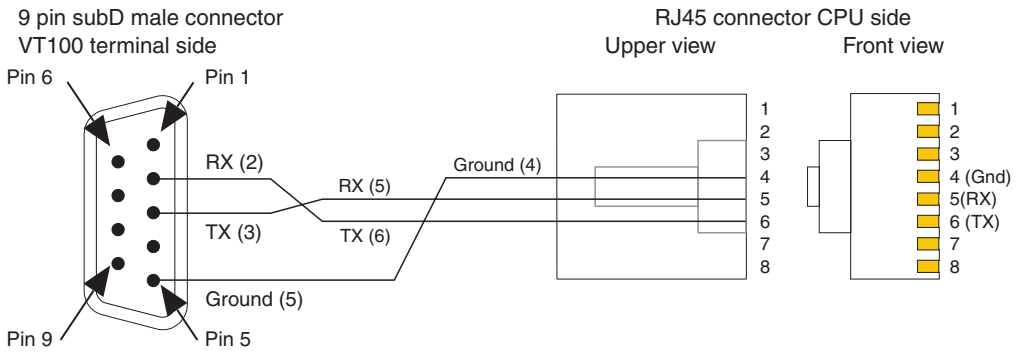
- RS232 communications protocol with RJ45 connector;
- Serial communications protocol with USB Type B connector.

In this way, depending o the terminal used to setup the CPU, the user must:

- Provide the proper connection cable;
- Set the correct communications parameters.

### 3-1-1 RS232 Serial Communications Connection

In order to connect a VT100 terminal or a PC with Hyper Terminal program, install an RS232 cable having the following characteristics:




*Setting the communications parameters*

At this point the user must configure the HyperTerminal in order to communicate with the correct COM Serial port. The number of the COM port linked to the USB connector can be found in:
*Start\ControlPanel\System\Hardware\Peripherals\Ports (COM and LPT)*
Using the COM port number open a new session of HyperTerminal and set the default communication parameters for the service port:

| | |
|---|---|
| **Baud rate** | 9600 |
| **Data** | 8bit |
| **Stop bit** | 1 |
| **Parity** | None |
| **Flow Control** | None |

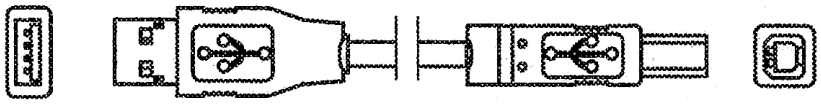
### 3-1-2 USB Serial Communications Connection

*Installing the cable*

In order to connect a Personal Computer using the USB (optional) connection, install an USB standard cable having an USB type “B” connector (CPU side) and an USB type “A” connector (PC side). This type of cable is standard for the USB connections and the installation is easy as each connector has only one insertion versus.

PC side
USB Type A Connector

CPU side
USB Type B Connector



*Installing the USB driver*

1. In Internet at the address:
   http://www.ftdichip.com/Drivers/VCP.htm
   locate the FT232B device and download the latest driver for the Operating System installed on the PC.
2. Connect the USB cable between the CPU and the PC.
3. Install the 2 drivers as explained in the installation manual, first the one of the USB device, then the driver that links the USB port to a COM port in the Personal Computer.

*Setting the communications parameters*

Once connected the CPU to the PC with the USB cable, the user must configure the HyperTerminal in order to communicate with the USB port. The number of the COM port linked to the USB connector can be found in:
*Start\ControlPanel\System\Hardware\Peripherals\Ports (COM and LPT)*
(locate the “*USB to COM #*” entry). Using the COM port number open a new session of HyperTerminal and set the default communication parameters for the service port:

| | |
|---|---|
| **Baud rate** | 9600 |
| **Data** | 8bit |
| **Stop bit** | 1 |
| **Parity** | None |
| **Flow Control** | None |

### 3-1-3 Setting the Communications Parameters of the CPU RS232 Service Port

The default configuration for the Service RS232 port is:

| | |
|---|---|
| **Baud rate** | 9600 |
| **Data** | 8bit |
| **Stop bit** | 1 |
| **Parity** | None |
| **Flow Control** | None |

During the configuration session it will be possible to change the baudrate, stop bit and parity. When the communications parameters of the CPU are changed, the communications parameters of the terminal (or PC) must be changed accordingly. The setup is performed by browsing the menus sent to the VT100 terminal or to the terminal emulation program (HyperTerminal) by the CPU.

### 3-1-4 Entering the Setup Menu

The two types of CPU (RS232 and USB) have activation modes of CPU setup program that differ for few actions.

- For a RS232 service port CPU, it is enought to run the HyperTerminal program using the correct configuration file and pressing the connect button.
- In case an USB service port CPU, after having installed and configured the USB driver and HyperTerminal, the user must:
  1. Run the HyperTerminal program using the correct configuration file;
  2. Power on the sigma**due** CPU;
  3. Wait for the audio notification that means that the Personal Computer has recognized and installed the “USB to COM” Driver;
  4. Press the connect button (☎).

*Accessing the Main Menu*

This procedure is valid for both the types of CPU (RS232 and USB). To start the setup session, press the ENTER (the PC sends a CR character to the CPU) key on the setup terminal **while CAN and ERR LEDs are blinking on the CPU**. If the character CR is not sent before a predefined time (start-up timeout) the system ends the configuration session and starts the PLC application. In this chapter some templates of a configuration session made with Windows Hyper Terminal are shown. After the reception of the first CR character, the welcome screen appears.




Press ENTER again to reach the configuration session Main Menu.

*Main Menu*

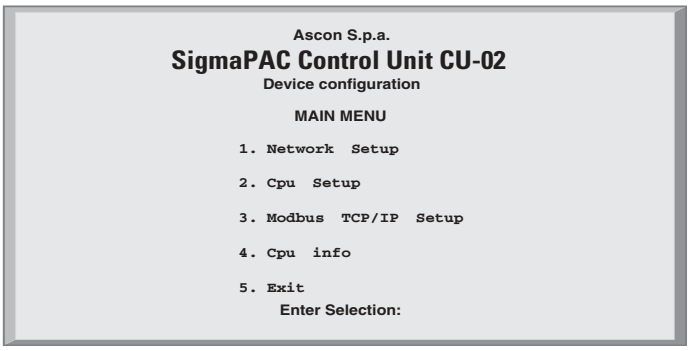



*Figure 3.1 - CPU configuration Main Menu*

Please note that the system has a further timeout that controls the life of the configuration session; this is the **inactivity** timeout. If the user does not work with the console for a time greater than this timeout, the configuration session will be automatically stopped and the PLC application will be started.
Both the described timeouts can be set during the configuration. The user should not set too short timeouts, thus preventing the possibility to work with the configuration console. To select an item of a menu or to insert a value for a parameter, the user has to type the corresponding number and than press ENTER.

## 3-2 Main Menu

As in *Figure 3.1 - CPU configuration Main Menu*, the Main menu has four sub menu selections:

**Network Setup:** To set the parameters of all the communication ports in the CPU;

**CPU Setup:** To set some parameter of the CPU;

**Modbus TCP/IP Setup:** Where the user can set the "*priority connection pool*" and the "*security address pool*". Moreover the user can set the number of messages per cycle and the "*broken connection timeout*".

**CPU info:** To see the release values of the FW, HW of the actual CPU;

**Exit:** To exit this menu and end the configuration session.

### 3-2-1 Network Setup Menu

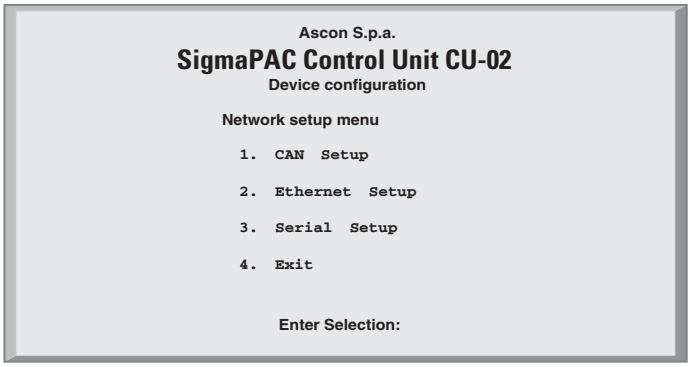



*Figure 3.2 - Network Setup Menu*

**CAN Setup:** To set the parameters for the CAN port;

**Ethernet Setup:** To set the parameters for the Ethernet port;

**Serial Setup:** To set the parameters for the RS232 port;

**Exit:** To exit this menu and return to the previous menu.

### 3-2-2 CPU Setup Menu

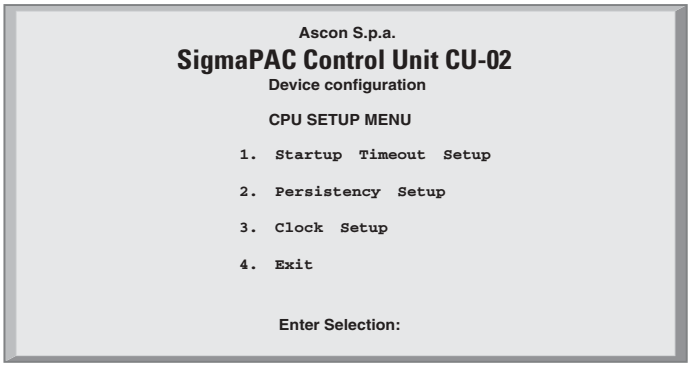



*Figure 3.3 - Figure 5 - CPU Setup Menu*

**Startup Timeout Setup:** To set the startup timeouts;

**Persistency Setup:** To set the parameters that control the saving of PLC program in the retentive memory;

**Setup CLOCK:** To set the parameters of the CPU real time clock;

**Exit:** To exit this menu and return to the previous menu.

### 3-2-3 CAN Setup Menu

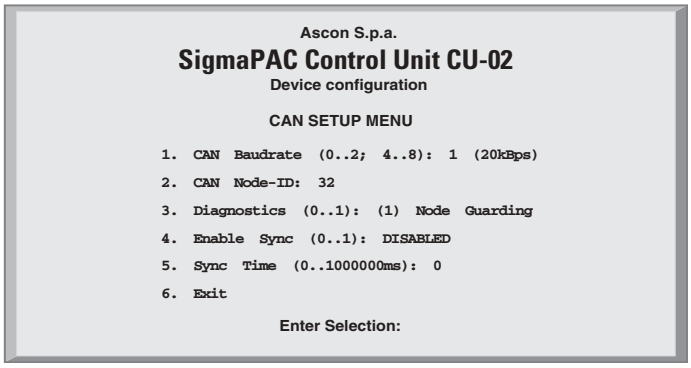



*Figure 3.4 - Figure 6 - CAN Setup Menu*

**CAN Baudrate**:To set the baud rate for the CAN network. The correspondence between the baud rate and the number of this item is indicated in Table 3.1 - CAN Network speed (note).

| VALUE | 0 | 1 | 2 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Speed kbps | 10 | 20 | 50 | 125 | 250 | 500 | 800 | 1000 |

*Table 3.1 - CAN Network speed*

**CAN node ID:** To set the Node-ID of the CPU in the CAN network;

**Diagnostic:** It is possible to work with Node Guarding protocol (default) that allows the CPU to receive a message from every node that is in the network or it is possile to select the nodeguarding proocol (1) in order to set the CPU to send and receive an "*Alive message*" from every node of the network and let the CPU and the nodes to understand if they are working well or there is a problem.

**Enable Sync:** It is possible to syet the Sync communications (process data objects on the CAN network are exchanged when a sync message occours).

**Sync time:** Generation time in millisecond of the sync message.

**Exit:** To exit this menu and return to the previous menu.

To change the value of an item the user must select the item number pressing the correnspondig item digit on the keyboard and press ENTER. The screen will then change, asking the user to "*Insert new value:*".

**Note:** Value 0 (10kbps) is not currently supported by sigma**due** I/O modules.

### 3-2-4 CPU Info Menu

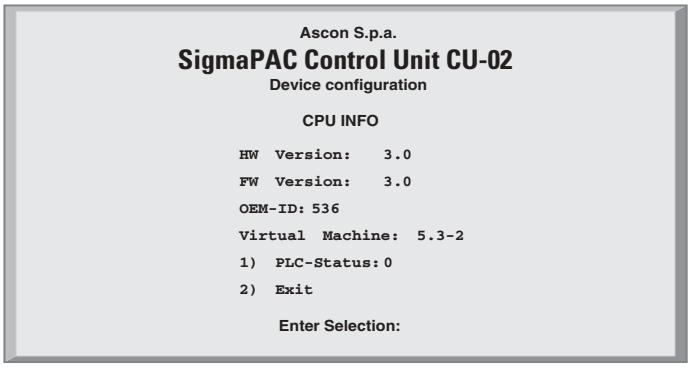



*Figure 3.5 - CPU Info*

**HW version:** Revision of the CPU Hadrware;

**FW version:** Revision of the CPU Firmware;

**OEM ID:** ASCON code for the runtme Software;

**V.MACHINE:** Version of the runtime Software (*5-3-2*)

**PLC-Status:** In this field is pointed out the CPU status:

| VALUE | Meaning |
|---|---|
| 0 | Normal status |
| 1 | Data configuration error |
| 2 | Retain error |
| 3 | Data configuration error + Retain error |
| 4 | Battery low |
| 5 | Battery low + Data configuration error |
| 6 | Battery low + Retain error |
| 7 | Battery low + Retain error + Data configuration error |

*Table 3.2 - Error codes that can be displayed in "CPU Info" screen.*

Entering **1** and pressing the return key, the user acqnowledges the alarms active displayed by "CPU Info" screen.

**Exit:** To Exit this menu and return to the previous menu.

### 3-2-5 ETHERNET Setup Menu

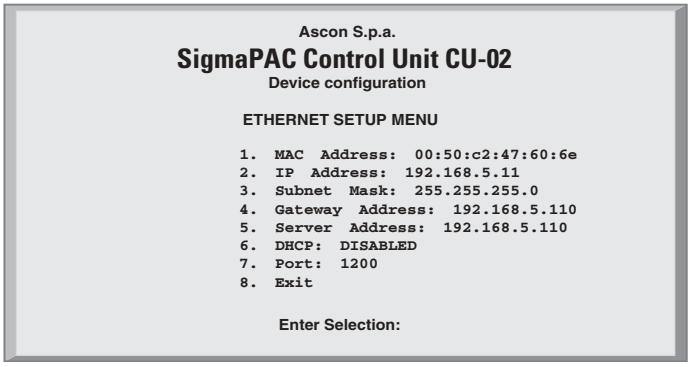



*Figure 3.6 - Ethernet Setup Menu*

**MAC Address:** To display the MAC address;

**IP Address:** To set the IP Address of the device;

**Subnet Mask:** To set the subnet mask of the device;

**Gateway Address:** To set the Gateway Address of the device;

**Server Address:** To set the Server Address of the device;

**DHCP:** To enable or disable DHCP Client on the device;

**Port:** To set the Port for the connection with OpenPCS;

**Exit:** To exit this menu and return to the previous one.

To set items 2, 3, 4 and 5, the user must digit on the keyboard the menu item number and press ENTER. The screen will then change, asking the user to "*insert new value:*". When the system displays the "*insert new value:*" message the user can enter the new values in number and dot notation.
To set DHCP value select item 6 and set 0 to DISABLE or 1 to ENABLE. To set the Port, select item 7 and enter the desired value.

**WARNING** Please note that items 2, 4 and 5 **MUST** have the same subnet in order to avoid communications problems.

### 3-2-6 Serial Setup Menu

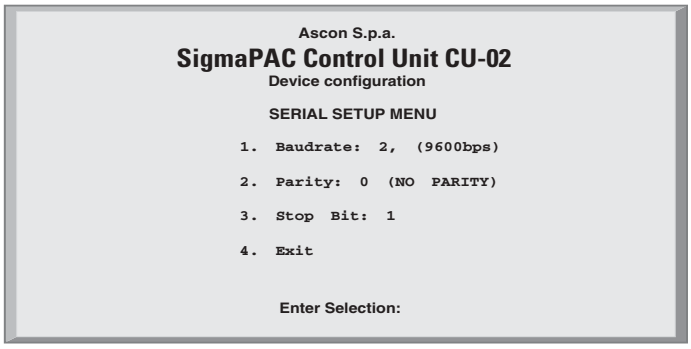



*Figure 3.7 - Serial Setup Menu*

**Baudrate:** This choice allows the user to set the baud rate for the RS232 communications port. The correspondence between the baud rate and the item number is specified in Table 3.3 - Serial Port Speed:

| VALUE | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| Speed bps | 2400 | 4800 | 9600 | 19200 | 38400 | 57600 | 115200 | 230400 |

*Table 3.3 - Serial Port Speed*

**Parity:** To set the parity for the RS232 communication port, according to Table 3.4 - Parity;

| VALUE | 0 | 1 | 2 |
|---|---|---|---|
| Parity | NONE | EVEN | ODD |

*Table 3.4 - Parity*

**Stop Bit:** To set the number of stop bits for the RS232 communications. Valid values are1 or 2;

**Exit:** To exit this menu and return to the previous one.

To change the value of an item, the user has to select the item number and press ENTER. The screen will then change, asking the user to *"Insert new value:*".

### 3-2-7 CPU Setup Menu

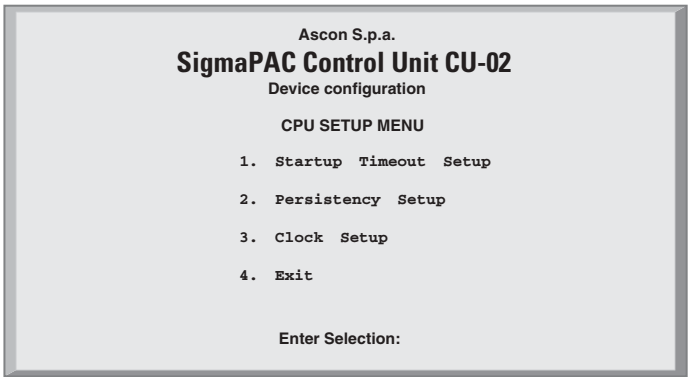



*Figure 3.8 - CPU Setup Menu*

**Startup Timeout Setup:** To set the startup timeouts;

**Persistency Setup:** To set the parameters that control the saving of PLC program in the retentive memory;

**Setup CLOCK:** To set the parameters of the CPU real time clock;

**Exit:** To exit this menu and return to the previous menu.

### 3-2-8 STARTUP Setup Menu

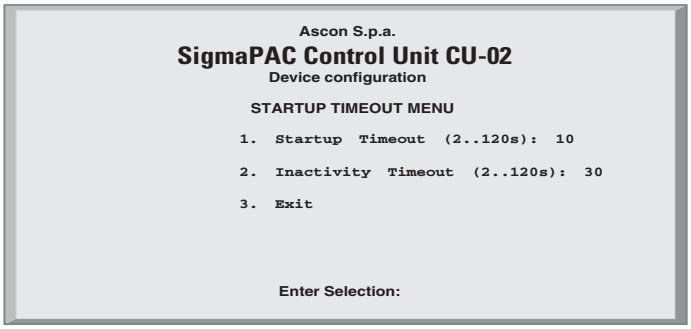



*Figure 3.9 - Startup Setup Menu*

**Startup Timeout:** To set the Startup Timeout;

**Inactivity Timeout:** To set the Inactivity Timeout;

**Exit:** To exit this menu and return to the previous one.

To change the value of an item, the user has to select the item number and press ENTER. The screen will then change, asking the user to "*insert new value:*".

### 3-2-9 Persistency Setup Menu

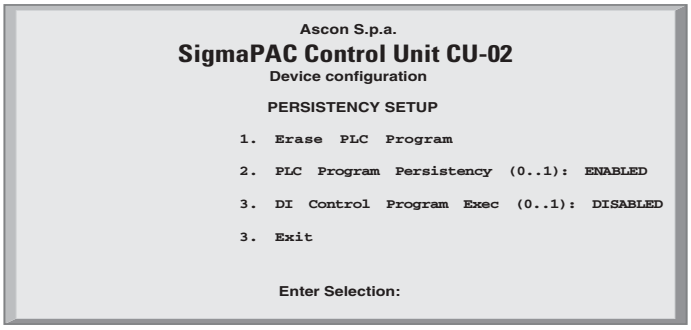



*Figure 3.10 - Pesistency Setup Menu*

The CPU is able to save a PLC program in persistent memory. This means that, when PLC Program Persistency is *ENABLED*, the program will be automatically loaded and executed at start-up.

Every time the user, during the development activities, downloads a new program to the CPU, this is saved in the persistent memory so, at next device start up, the last downloaded program will be executed.
Saving a program in persistent memory is a time consuming activity. This wait time can be annoying during program development and debugging. For that reason the user may want to disable the automatic program save in persistent memory, making development activities more efficient.
Sometimes can be useful to prevent executing any program at startup time.
Selecting the item "*Erase PLC program*" the retentive memory area reserved to store PLC programs is erased. This activity take some seconds. When the "*Persistency setup menu*" screen reappears then the memory has been erased.

**Erase PLC Program:**
Selecting the item "*Erase PLC program*" the retentive Memory area reserved to store PLC programs is erased;

**PLC Program Persistency (0..1):**
The item "*PLC program persistency*" allows the user to configure the program store memory to be retentive or volatile memory;

**DI Control Program Exec (0..1):**
The Digital Input Control Program Execcan be used as a generic input addressed as % ±x0.0 (default) or can be set as a START (1) or STOP (0) contact for the actual program that is downloaded on the CPU.

**Exit:** To exit this menu and return to the previous one.

### 3-2-10 CLOCK Setup Menu

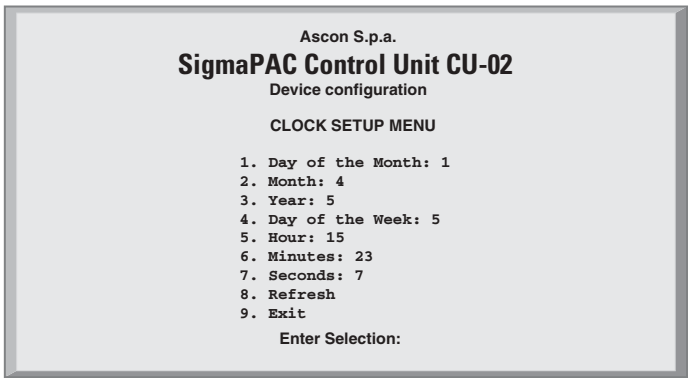



*Figure 3.11 - Clock Setup*

### 3-2-11 Modbus TCP/IP Setup

Selecting item 3 on the Main Menu you will arrive at the following menu:

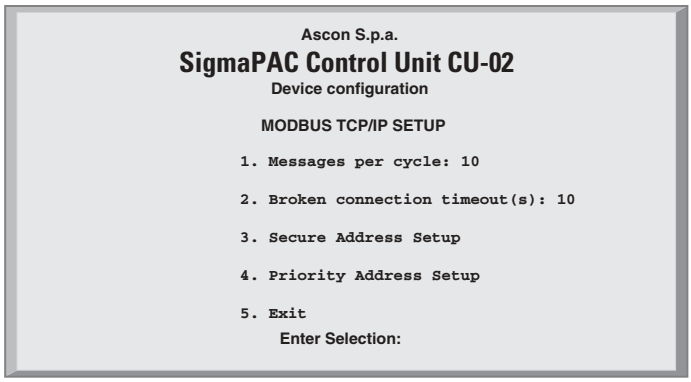



*Figure 3.12 - Modbus TCP/IP Setup Menu*

**Messages per Cycle:**
By the first item the user can insert the number of messages processed during each cycle. The inserted value must be comprised between 1 and 50.

**Broken Connection Timeout (s):**

By the second item you can set the inactivity timeout of a TCP/IP connection (in seconds). The inserted value must be comprised between 10 and 5400. To verify the connection state after a long period of inactivity, the TCP/IP “keepalive” protocol is used. The protocol performs in sequence the following steps:

1. At each received messages the timeout is zeroed;
2. In the case that the programmed timeout is reached, a “probe” message is sent in order to verify if the connection is still active;
3. If an answer to the “probe” is received, then the timeout is zeroed.
4. In case of no answer, the “probe” will be sent again three times, every 10 seconds.
5. After the fourth “probe” has got no answer the connection will be closed.

**Secure Address Setup:**

Selecting this item a new window will appear displaying the menu called “Secure Address Protocol”(see next);

**Exit:** exit this menu and return to the previous menu.

## 3-2-12 Modbus TC/IP Secure ADDRS Table Menu

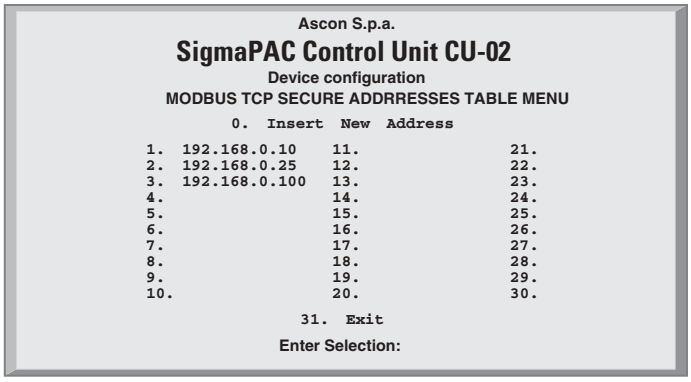



*Figure 3.13 - Modbus TCP/IP Secure Address Table Menu*

By this menu, in the case the security functions are enabled, the addresses that the Modbus TCP/IP Server can access are set out.
To insert a new address, select 0) first, then type in the new address; it will be inserted in the first free position.
To delete an address, select the number of the address you want to remove.

## 3-2-13 Modbus TC/IP Priority ADDRS Table Menu

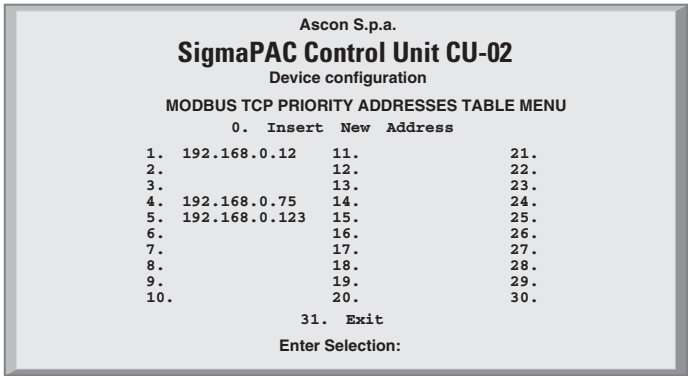



*Figure 3.14 - Modbus TCP/IP Secure Address Table Menu*

The insertion rules are the same as described for the “*security address pool*”.

Addresses inserted in the "*priority connection pool*" are managed by the system in a privileged way. The Modbus TCP/IP server agent can maintain up to 10 TCP connections at the same time. When a new connection request arises and all available connections are engaged, the system shall make a choice.
Since every new request must be satisfied, one of the present active connections shall be closed.
The first candidates that are to closed are the addresses not belonging to the "*priority connection pool*". The second choice is to close the connection not active for the greater amount of time.

# *Chapter 4*
# *Programming the CPU*

## 4-1 Installing OpenPCS

### 4-1-1 Hardware and Software Requirements

As a programming system, you will need an IBM compatible PC with at least a Pentium microprocessor (we recommend a Pentium 4 with 1GHz of clock frequency) and a CD-ROM-drive; a hard disk drive with at least 100Mbytes of free space; a minimum total amount of 512Mbytes (recommended) of memory, a high-resolution graphics card and an ethernet 10 baseT ethernet adapter. Besides OpenPCS, you will need an operating system, which can be Windows 2000 or Windows XP and Windows Server 2003.

### 4-1-2 Installation

OpenPCS is delivered on CD-Rom. The CD auto-starts a screen where you can select the software you want to install. If auto-start is not activated or does not work, please start the **`PS544P.EXE`** from the subdirectory:

`X:\SETUP\`

Where X: is the letter assigned to the CD-ROM drive in your PC.

When installing OpenPCS, you will be asked to enter your licence information. If you do not have a licence, just keep the serial number and key fields empty. Without a licence OpenPCS is still full functional, but restricted to the "SIMULATION". Notice that in Simulation mode a different target should be selected as "HW modules" and it is not posible to use many of the fiunction blocks available in the ASCON libraries.

### 4-1-3 Starting OpenPCS

Start Windows and choose:

***Start → Programs → infoteam OpenPCS 2006 → infoteam OpenPCS 2006***

in the start-menu; this will open the ControlX-Framework.

### 4-1-4 Configuring OpenPCS

Before you can start programming with OpenPCS you should configure it, in order to work with the ASCON CPU target. A **`.cab`** file must be installed into OpenPCS. The file **`ascon_OPCS54_HWvvxx_LIByy_HELPzz.cab`** contains a series of files describing Ascon sigma**due** Hardware, drivers, examples and utilities (`vv`, `xx`, `yy` and `zz` are digit to identify the software version)
In the OpenPCS "*Extras*" menu, select "*tools – Driver install…*". "*Select*" the desired cabinet (e.g. ascon_OPCS54_HW3030_LIB01.cab), then "*Install*".

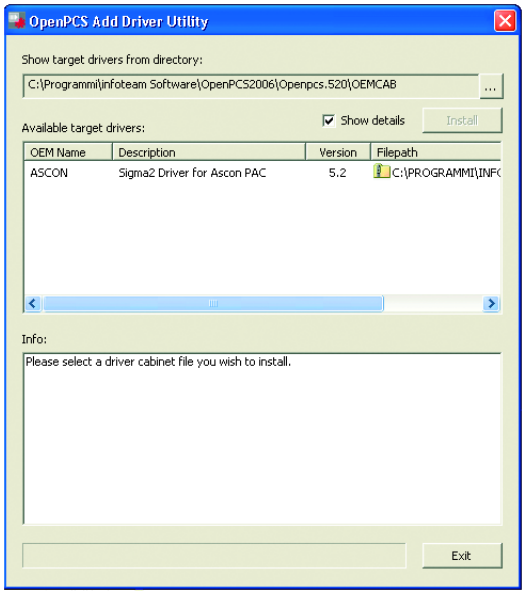



*Figure 4.1 - OpenPCS OEM Driver Installation*

## 4-2 OpenPCS Setup

To connect the OpenPCS development system to the Ascon target, a new connection must be defined.
Select "*Connections...*" item in the "*PLC*" menu. In the window of *OpenPCS Connection Setup* select "*New*".
Now in the window "*Edit connection*" it is possible to set the new connection. In the field "*Name*" you can write the reference name of the new connection.
Pushing the "*Select*" button it is possible to pick the driver that manage the communication with the target, for Ascon CPU is TCP52.

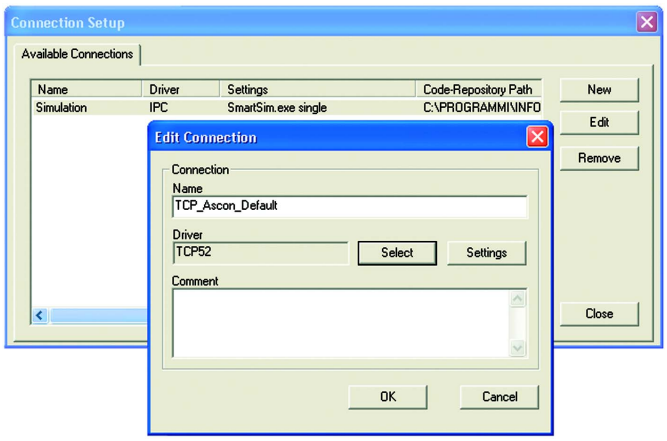



*Figure 4.2 - OpenPCS Connection Setup*

Now, clicking on the "*Settings*" button it is possible to set the communication parameters.

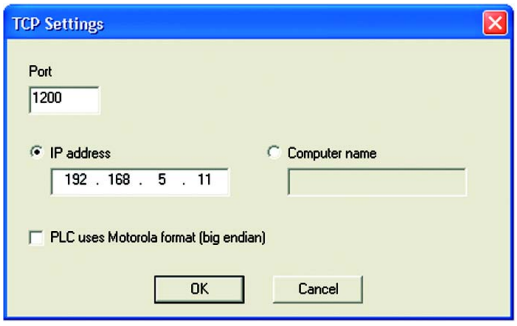



*Figure 4.3 - TCP Settings*

The Port number and IP address must be the same configured at the initial CPU configuration session. See the Ethernet setup menu, items 7 and 2.
OpenPCS environment is now ready to communicate with the Ascon target.
To use the CPU is mandatory to set up the project.
Select the "*Resource Properties*" item in the PLC menu, set "*Ascon…*" in the "*Hardware Module*" field and the newly created TCP connection in the "*Network Connection*" field.

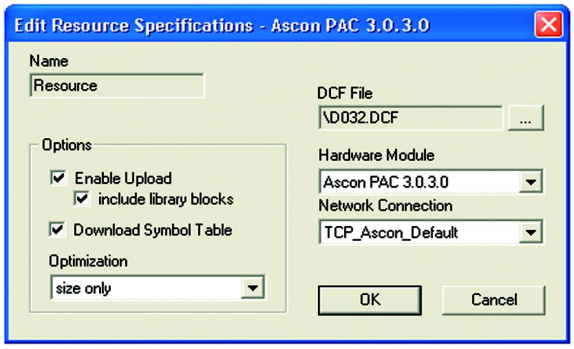



*Figure 4.4 - OpenPCS resource Specifications*

OpenPCS supports uploading of projects from your controller. To enable this feature, the "*enable upload*" box has to be checked when compiling a resource.
The code "*Optimisation*" menu allows for three choices of compilation: "*Normal*" and "*Speed only*" refers to the NCC: Native Code Compilation, while "*Size only*" refers to the standard code.
Please note that the use of NCC doesn't permit to insert break points in debugging projects.

*Setup Communication Timeout*

Depending on several reasons it may be necessary to set the communication timeout on Ethernet Port to values higher than the default value. This timeout checks the dialogue between OpenPCS and the target CPU. If you are going to deal with large programs, the compilation time may be excessively long. The default value of 20000ms can be increased at registry key:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\infoteam Software GmbH\
OpenPCS\5.4.4\Online\TcpDriverTimeout_ms]
```

Value = "20000" means a timeout of 20 seconds.

## 4-3 Software Utilities

### 4-3-1 OPC Server

An OPC Server is available onto OpenPCS programming suite. In order to have access to the OPC services, the OpenPCS complete suite should be installed on the PC used for data exchange.
OPC means OLE for Process Control and is a series of standards specifications created by the OPC Foundation.
OpenPCS currently supports the Data Access Specification (OPC-DA) Version 2.0. For further information on OPC, please consult the web page of the OPC Foundation at:
**`http://www.opcfoundation.org/`**

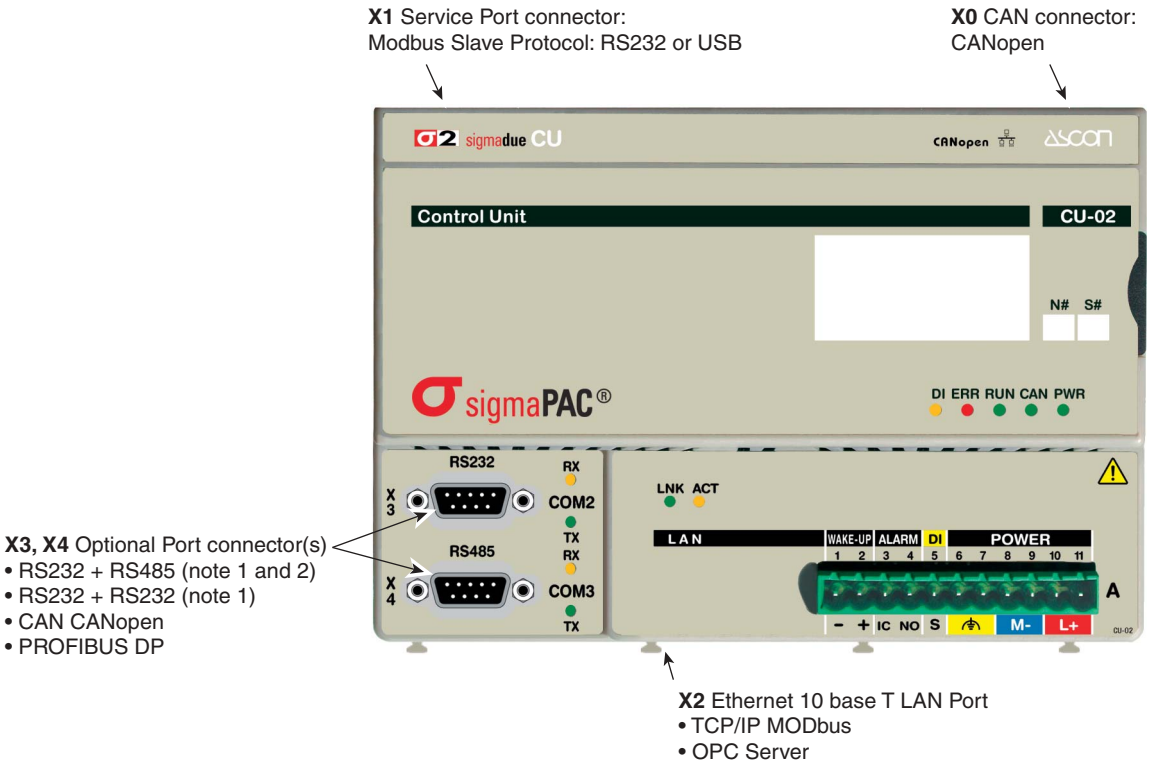
The OPC server will take all information regarding OPC tags and communication information from a database. To use OPC, you have to do nothing. The database is automatically created with each compilation from the resource global variables. This database can be found at:
**`<project root directory>\$ENV$\RESOURECE\OpcSvr.mdb`**

It is recommend not to change that file.

## 4-4 Communication Ports Protocols

sigma**due** CPU has various communication ports and protocols. The combinations of ports and protocols are shown below:

*Figure 4.5 - Communication Ports Protocols*




**Notes:** **1.** Modbus Master/Slave
**2** Consult the Installation Guide to polarise and terminate the RS485 port.

## 4-5 Auxiliary digital port

The auxiliary digital port will be available for a PLC project, provided that the resource is mapped on memory area that the project can see. The port is mapped as follows:

| Port | Mapping |
|---|---|
| WAKE-UP output | %QX0.0 |
| Alarm Output | %QX0.1 |
| General purpose digital input | %IX0.0 |

## 4-6 Watchdog Timer

Ascon CU-02 CPU contain a watchdog control, controlled by 2 specific FBs. Watchdog is fundamentally a down counter, decremented in 100ms intervals. When the count value reaches zero, two different behaviours of the system may occur, depending on how the configuration is:

- PLC program keeps working, stores the event and force the Alarm relay to OPEN;
- PLC program performs a Reset and starts execution again.

Please note that the Watchdog timer is controlled by FB but it runs independently of the PLC program. Therefore, if the program is stopped, e.g. during a debug session, the timer keeps running, and behaves as programmed when the count reaches zero.
During debug sessions it is advisable to maintain the Watchdog function disabled.

## 4-7 CANopen Extension for Ascon sigma**due** CPU

### 4-7-1 Introduction

sigma**due** is a system based on two main concepts: the CANopen technology and the IEC61131 standard languages. The integration of these two different worlds is performed by:

- Use of networked variables;
- Direct access to CANopen parameters and functions via predefined low level communication function blocks. CANopen services for CPU programs according to the IEC61131-3 standard are defined in the CiA (CAN in Automation e.V.) Draft Standard 405;
- Use of Ascon IO Library.

*Using Networked Variables*

Networked variables are the easiest way of data exchange in a CANopen network system. Within the CPU, program access to the network variables occurs in the same way as access to internal, local variables on the CPU. From the point of view of a CPU programmer it is unimportant whether a input variable is assigned to a local input on the CPU device or to an input on a networked expansion device. The use of networked variables only requires basic knowledge of CANopen. In general, a CANopen configuration tool as well as the availability of EDS files for the individual CANopen devices are required for integrating network variables into a CPU.

With the help of network variables it is possible:

- to expand the number of inputs and outputs on the PLC using distributed CANopen I/O devices;
- to exchange process data between various control units, thus realizing decentralized automation control projects;

- to integrate other special CANopen devices from third party vendors into a complex control project and furthermore to build very specific automation control systems using modular components in addition to Ascon sigma**due** CANopen IO modules.

For a detailed description see **[3]** and **[4]**.

*Using CANopen Function blocks*

CANopen function blocks enable direct access to specific CANopen services, thus offering a high degree of flexibility in the target application. Furthermore, using these function blocks does not require an additional CANopen configuration tool or EDS files. However, using the CANopen function blocks assumes that the user has detailed knowledge about CANopen and its services.

With the help of CANopen function blocks it is possible:

- to directly exchange data with other CANopen nodes via SDO (Service Data Object) or PDO (Process Data Object);
- to enquire and change the state of other CANopen nodes;
- to receive error messages from other CANopen nodes;
- to allow generation of SYNC messages.

For a detailed description see **[3]** and **[4]**.

*Using Ascon I/O Library*

The function blocks present in the Ascon IO Library enable the access to Ascon sigma**due** CANopen IO modules, providing services that allow the CPU programmer to do not deal with the fieldbus directly. To provide this level of abstraction, the function blocks are based on those described above.

With the help of the function blocks present in the library is possible:

- to directly exchange data with all sigma**due** CANopen module;
- to setup all the module's functionalities;
- to enquire and see the state of the modules;
- to reset the module.

For a detailed description see **[11]** and **[12]**.

# *Chapter 5*
# *Ascon Function Blocks Libraries*

In this chapter are listed all the libraries present in the Ascon installation of the OpenPCS programming tool, and all the function blocks present in the firmware of the CU-02 device. For each library is also indicated the complete list of the function blocks with a little description. For more details please refer to the specific dedicated documentation.

## 5-1 AsconACLib

The *AsconACLib* is a function block library that contains a set of generic functionalities that come from the Ascon AC Station Device useful for the IEC 61131 programming (see the "*IEC 61131-3 Function Block Library*" [**11**] manual for details).
The table here reported gives the complete list of the function blocks of the library.

| Function Block name | Description |
|---|---|
| ALARM_ABS | Absolute Alarm Function Block |
| ALARM_ADVANCED | General Alarm Function Block |
| ALARM_BND | Band Alarm Function Block |
| ALARM_DEV | Deviation Alarm Function Block |
| ALARM_RATE | Rate Alarm Function Block |
| CHARACTERIZER_8 | Linear Interpolation with 8 points |
| CHARACTERIZER_16 | Linear Interpolation with 16 points |
| COMPARATOR | Comparator with hysteresis Function Block |
| CONV_AD8 | From BYTE to 8 bits |
| CONV_AD16 | From WORD to 8 bits |
| CONV_AD32 | From DWORD to 8 bits |
| CONV_DA8 | From bits to BYTE |
| CONV_DA16 | From bits to WORD |
| CONV_DA32 | From bits to DWORD |
| COUNTER | Rising Edge Counter |
| DECODER_8 | Decoder Function Block |
| FLIPFLOP_D | D Type FlipFlop Function Block |
| FLIPFLOP_JK | JK Type FlipFlop Function Block |
| HOLD_VALUE | Sample & Hold Function Block |
| INBETWEEN | Middle Selector Function Block |
| LIMITER_VALUE | Limiter Function Block |
| MIN_MAX_SELECTOR | Min/Max Selector Function Block |

| Function Block name | Description |
|---|---|
| MONOSTABLE_DS | Monostable with Delay |
| MONOSTABLE_NED | Monostable with Delay on the Negative Edge |
| MONOSTABLE_PED | Monostable with Delay on the Positive Edge |
| MONOSTABLE_PUL | Monostable Pulse Generator |
| MUX_A8 | Analog Multiplexer 8 Input |
| MUX_A16 | Analog Multiplexer 16 Input |
| MUX_D8 | Digital Multiplexer 8 Input |
| MUX_D16 | Digital Multiplexer 16 Input |
| RESCALE | Rescaling Function Block |
| SLOPE_LIMIT | Slope Limiter |
| TOTALIZER | Totalizer Function Block |

## 5-2 AsconBasicIOLib

The *AsconBasicIOLib* is a Function Block library that allows the *OpenPCS* programming environment to access di Ascon's sigma**due** devices.
These FBs free the user from controlling the communications details of CANopen[1] protocol. Using these FBs the user simply accesses the communications basic functions of each I/O module (see the "*IEC 61131-3 Function Block Library*" [**11**] manual for details).
A list of available I/O FBs follows:

| | |
|---|---|
| bDI16LV | FB to interface the module DI-16LV |
| bDI32LV | FB to interface the module DI-32LV |
| bDO04RL | FB to interface the module DO-04RL |
| bDO08RL | FB to interface the module DO-08RL |
| bDO04TX | FB to interface the module DO-04TX |
| bDO16TS | FB to interface the module DO-16TS |
| bDO16TP | FB to interface the module DO-16TP |
| bDO32TS | FB to interface the module DO-32TS |
| bDM08TS | FB to interface the module DM-08TS |
| bDM16TS | FB to interface the module DM-16TS |
| bDM32TS | FB to interface the module DM-32TS |
| bAI02UI | FB to interface the module AI-02UI |
| bAI04RT | FB to interface the module AI-04RT |
| bAI08TC | FB to interface the module AI-08TC |
| bAI08HL | FB to interface the module AI-08HL |
| bAI08DP | FB to interface the module AI-08DP |
| bAO08HL | FB to interface the module AO-08HL |
| bAO08DP | FB to interface the module AO-08DP |
| bERRORSTATEAN | FB to manage the communications error statuses for analogue output modules |
| bERRORSTATEDIG | FB to manage the communications error statuses for digital output modules |

## 5-3 AsconControlLib

The *AsconControlLib* is a function block library dedicated to the process control. It uses the basic functionalities dedicated to the PID implementation present in the firmware of the control unit (CU-02) device in order to provide solution ready to use. In fact in the library there is the implementation of a complete standard regulator in both versions: single action and double action for heat and cool application. Please note that are present also different function blocks dedicated to the tuning algorithms in both version “Step Response” and “Natural Frequency”.
The table here reported gives the complete list of the function blocks of the library (see the “*IEC 61131-3 Function Block Library*” [**11**] manual for details).

| | |
|---|---|
| S2_CONTROLLER | Single Action Controller |
| S2_FILTER | First Order Filter |
| S2_HC_CONTROLLER | Heat and Cool Controller |
| S2_HC_TFUZZY | Tuning with Fuzzy Logic for Heat and Cool Loops |
| S2_HC_TNATFREQ | Tuning with Natural Frequency Algorithm for Heat and Cool Loops |
| S2_HC_TSTEPRESP | Tuning with Step Response Algorithm for Heat and Cool Loops |
| S2_TFUZZY | Tuning with Fuzzy Logic for Single Action Loops |
| S2_THHA | Tuning with Modified Step Response Algorithm for Single Action Loops |
| S2_TNATFREQ | Tuning with Natural Frequency Algorithm for Single Action Loops |
| S2_TSTEPRESP | Tuning with Step Response Algorithm for Single Action Loops |

## 5-4 AsconCPULib

The *AsconCPULib* is a function block library that allows the access to the control unit (CU-02) device of the Ascon sigma**due** line, from the OpenPCS programming tool. These FBs allow the user to set and manage the CANopen network activities: diagnostic, failure management of the connected devices, syncronization,…
The table here reported gives the complete list of the function blocks of the library. Please note that some of these must be considered as *system function blocks:* in fact these FBs implement particular functionalities dedicated to the global managment of the CANopen network (see the “*IEC 61131-3 Function Block Library*” [**11**] manual for details).

| Function Block name | Description |
|---|---|
| S2_CU02 | Function block interface for the Control Unit Module |
| SPLIT_ENABLE | Extract the enable signal from the Network connection of the modules |
| SET_TT **[note]** | Set the communication type of a CANopen node |
| SET_TT_MODULE **[note]** | Set the transmission type parameters of all PDOs of a device |
| RECOGNIZESIGMAIO **[note]** | Recognize the module name of a device coming from the sigma**due** IO line |

ï ï ïï ï These are system Function Blocks

## 5-5 AsconMBCommLib

The *AsconMBCommLib* is a Function Block library that allows a simplified access to the communications port of Ascon's CU-02 sigma**due** CPU (see the "*IEC 61131-3 Function Block Library*" [**11**] manual for details). A list of FBs available in the library follows:

| | |
|---|---|
| MB_MST_RD_COIL | Modbus Master: Read Coil |
| MB_MST_WR_COIL | Modbus Master: Write Coil |
| MB_MST_SYNC | Modbus Master: Syncronization |
| MB_MST_RD_WORD | Modbus Master: Read Word |
| MB_MST_WR_WORD | Modbus Master: Write Word |
| MB_SLV_RD8_DWORD | Modbus Slave: Read 8 DWORD values |
| MB_SLV_RD8_REAL | Modbus Slave: Read 8 REAL values |
| MB_SLV_RD16_WORD | Modbus Slave: Read 16 WORD values |
| MB_SLV_RD32_DIGITAL | Modbus Slave: Read 32 digital values |
| MB_SLV_RD_DIGITAL | Modbus Slave: Read 1 digital value |
| MB_SLV_RD_DWORD | Modbus Slave: Read 1 DWORD value |
| MB_SLV_RD_REAL | Modbus Slave: Read 1 REAL value |
| MB_SLV_RD_WORD | Modbus Slave: Read 1 WORD value |
| MB_SLV_WR8_DWORD | Modbus Slave: Write 8 DWORD values |
| MB_SLV_WR8_REAL | Modbus Slave: Write 8 REAL values |
| MB_SLV_WR16_WORD | Modbus Slave: Write 16 WORD values |
| MB_SLV_WR32_DIGITAL | Modbus Slave: Write 32 digital values |
| MB_SLV_WR_DIGITAL | Modbus Slave: Write 1 digital value |
| MB_SLV_WR_DWORD | Modbus Slave: Write 1 DWORD value |
| MB_SLV_WR_REAL | Modbus Slave: Write 1 REAL value |
| MB_SLV_WR_WORD | Modbus Slave: Write 1 WORD value |
| SERIAL_PORTS | Set the configuration for the ModbusRTU ports |
| TCP_IP_PORT | Set the configuration for the ModbusTCP port |

## 5-6 AsconLibrary

The AsconLibrary controls all the functions provided by the I/O modules of the sigma**due** series. It masks the fieldbus communications to and from the modules so that the user does not need to deal with the fieldbus directly: from explicit use of the protocol messages (SDO, PDO, EMCY, SYNC, …) to the use of the configuration files such as EDS and DCF (see the "*I/O Function Block Library*" [**12**] manual for details). The table here reported gives the complete list of the function blocks of the library.

| | | |
|---|---|---|
| Operator Panel with CANopen interface (slave) | WRITE_OP | Transfer from PLC and display a value on OP |
| | READ_OP | Transfer a value from OP to CPU |
| | UPDATE_OP | Keep updated a local copy of OP database |
| DI-16LV | DI_16LV_STD | Transfer 16 discrete inputs states from module to CPU |
| | DI_16LV_LTH_TIME | Manage the Input special functions of the module:<br>• Latched Inputs<br>• Monostable Inputs |
| | DI_32LV_STD | Transfer 32 discrete input states from module to CPU |

| | | |
|---|---|---|
| DO-04RL | DO_04RL_STD | Transfer 4 relay Outputs from CPU to the module |
| | DO_04RL_PULSE | Manage the Output special functions of the module:<br>• Pulse Output |
| DO-04TX | DO_04TX_STD | Transfer 4 discrete Outputs from CPU to the module |
| | DO_04TX_PULSE | Manage the Output special functions of module:<br>• Pulse Output |
| DO-08RL | DO_08RL_STD | Transfer 8 relay Outputs from CPU to the module |
| | DO_08RL_PULSE | Manage the Output special functions of the module:<br>• Pulse Output |
| DO-16TS | DO_16TS_STD | Transfer 16 discrete Outputs from CPU to the module |
| | DO_16TS_PULSE | Manage the Output special functions of module:<br>• Pulse Output |
| DO-16TP | DO_16TP_STD | Transfer 16 discrete Outputs from CPU to the module |
| | DO_16TP_PULSE | Manage the Output special functions of module:<br>• Pulse Output |
| DO-32TS | DO_32TS_STD | Transfer 32 discrete Outputs from CPU to the module |
| AI-02UI | AI_02UI_STD | Transfer 2 analogue Input from module to CPU |
| | AI_02UI_RESCALE | Scale and Transfer 2 linear analogue Inputs from module to CPU |
| AI-08HL | AI_08HL_STD | Transfer 8 analogue Input from module to CPU |
| AO-08HL | AO_08HL_FAST | Transfer 8 analogue Output from CPU to module in FAST mode |
| | AO_08HL_ENHANCED | Transfer 8 analogue Output from CPU to module in ENHANCED mode |
| | AO_08HL_RAMP | Manage the generation of 8 “ramp” type analogue Outputs |
| | AO_08HL_LINEARIZATIO N | Manage the “Linearization Table” of 8 custom analogue Outputs |
| | AO_08HL_RESCALE | Scale and Transfer 8 linear analogue Outputs from CPU to module |
| | AO_08HL_FAST_ENHAN CED | Select command for FAST / ENHANCED working mode of the module |
| DM-08TS | DM_08TS_STD | Select I/O channels and transfer discrete Input states from module to CPU and discrete Outputs from CPU to the module |
| | DM_08TS_PWM_PULSE | Manage the generation of “PWM“ AND pulse on ch # 3 and 4 |
| | DM_08TS_LTH_TIME | Manage the Input special functions of the module:<br>• Latched Inputs<br>• Monostable Inputs |
| | DM_08TS_FREQ_COUN T_PERIOD | Manage the Input special functions on ch# 1 and 2 of the module:<br>• Frequency input<br>• Period measurement<br>• Counter input |
| DM-16TS | DM_16TS_STD | Transfer 8 discrete inputs states from module to CPU and transfer 8 discrete Outputs from CPU to the module |
| DM-32TS | DM_32TS_STD | Transfer 16 discrete inputs states from module to CPU and transfer 16 discrete Outputs from CPU to the module |
| CAN MESSAGES | RPDO | Receive a PDO from the CANopen network |
| | WPDO | Send a PDO on the CANopen network |

## 5-7 Firmware Function Blocks List

In this section are listed the firmware function blocks that are present on the CU-02 (hardware version 3.0.3.0). For each of the function block here reported a short description is provided (see the "*Ascon Firmware Function Block Library*" [**10**] manual for details): for more details please refer to the specific documentation.

| | |
|---|---|
| OPEN_SERIAL_COMM | Configure the serial port and set the protocol used on it |
| MODBUS_SLAVE_SETTINGS | Set the node_id and timeout parameters of the MB slave agent |
| MODBUS_SLAVE_STATUS | Check the status of the MB agent |
| MODBUS_MASTER_EXECUTE | Execute a query in compliance with the MB protocol |
| MODBUS_MASTER_STATUS | Check the status of the MB agent. |
| MODBUS_GET_SLAVE_DATA | Read registers from a memory area dedicated to a MB slave |
| MODBUS_SET_WORD_DATA | Write registers to a memory area dedicated to a MB slave |
| MODBUS_SET_DWORD_DATA | Write two contiguous registers (4 bytes) to a memory area dedicated to a MB slave |
| MODBUS_GET_DIGITAL_SLAVE | Read 16 digital value from a memory area dedicated to a MB slave |
| MODBUS_SET_DIGITAL_SLAVE | Write 16 digital value to a memory area dedicated to a MB slave |
| CLOSE_SERIAL_COMM | Close the serial communication port |
| RTC_SETUP | Set the system clock |
| RTC_GET_VALUES | Read the system clock |
| WATCHDOG_SET | Configure the system watchdog |
| WATCHDOG_STATUS | Checking the status of the system watchdog |
| CTRL_PID | PID algorithm |
| CTRL_MV | Automan Station for single action regulation |
| CTRL_HCMV | Automan Station for heat and cool regulation |
| CTRL_SPLITMV | Automan Station for heat and cool regulation with split range |
| CTRL_SRV | Servomotors algorithm |
| CTRL_TPO | Time proportional output |
| MB_TCP_GET_CONN_BY_ADDR | Return information of a connection identified by the client IP address |
| ENABLE_MODBUS_TCP_SERVER | Set and activate the MBTCP/IP Server agent |
| CLOSE_MODBUS_TCP_SERVER | Disable MBTCP/IP Server |
| MB_TCP_CLOSE_CONN | Close one of the 10 active connections |
| MB_TCP_CONN_STATUS | Show the status of a MBTCP/IP connection |
| MB_TCP_GET_CONN_CONFIG | Return information of a connection |
| SERIAL_IO_CONFIG | Configure the ASCII serial port |
| SERIAL_IO_READ | Read data from the ASCII serial port |
| SERIAL_IO_WRITE | Write data on the ASCII serial port |
| RAND | Generete random numbers from 0 to 65535 |
| ASCON_FLATTEN_TO_REAL | Convert the 4 bytes of the input parameters as the flatten equivalent of a real number which is then output-returned |
| ASCON_REAL_TO_FLATTEN | Convert the REAL variables in their FLATTEN equivalents |

# Chapter 6
# Technical data

## 6-1 General conditions

| | |
|---|---|
| Power supply | 24Vdc –15...+25% |
| Power consumption | 5W |
| Dimensions | L: 152, H: 110, W: 65 mm |
| Weight | 400g |
| Isolation | 3-way isolation 800Vpp (see diagram) |

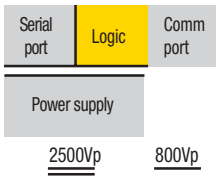



*Figure 6.1 - CPU isolation diagram*

## 6-2 Ambient conditions

| | |
|---|---|
| Regulations | Developed according to EN 61131-2, VDE 0160, UL 508 |
| Ambient temperature, operating | 0...+55°C |
| Ambient temperature, storage/transport | -25...+85°C |
| Temperature cycles (speed of changing from positive-to-negative and vice versa) | 0.5°K/min (no condensation) |
| Humidity, operating | 75% on average; 85% occasionally (no condensation).<br>**Note:** Ranging from 0°C to +55°C, appropriate measures must be taken against increased humidity (>85%) |
| Humidity, storage/transport | 75% average; 85% occasionally.<br>**Note:** For a short period, slight condensation may appear on the housing |
| Degree of protection according to IEC 60529 | IP20 |
| Degree of protection according to IEC 60536 | Class 3 |
| Housing material, basic | Plastic PVC-Free PA6 6, self-extinguishing (V0) |
| Degree of pollution according to EN 50178 | 2; condensation not permissible in operation |
| Surge voltage class | II (low-level signal); III (power level) |

# *Chapter 7*
# *Reference documents*


**[1]** OpenPCS 5.4.4 – User Manual.

**[2]** IEC 61131-3: Programming Industrial Automation Systems – Karl-Heinz John, Michael Tiegelkamp - Springer.

**[3]** CANopen Extension for IEC61131-3 – User manual – Edition March 2005 – Systec Electronic.

**[4]** CiA DS 405 V2.0: CANopen Interface and Device Profile for IEC61131-3 Programmable Devices.

**[5]** CiA 301 DSP V4.1: CANopen application layer and communication profile

**[6]** CU-02 Installation manual (code: J30 - 658 - 1ACU-02 E).

**[7]** CU-02 User manual (code: J30 - 478 - 1ACU02 E).

**[8]** sigma**due** I/O modules Installation Manuals: DI-16LV, DI-32LV, DO-04RL, DO-04TX, DO-08RL, DO-16TS, DO-16TP, DO-32TS, AI-02UI, AI-08HL, AO-08HL, DM-08TS, DM-16TS.

**[9]** sigma**due** I/O modules User Manuals: DI-16LV, DI-32LV, DO- 04RL, DO-04TX, DO-08RL, DO-16TS, DO-16TP, DO-32TS, AI-02UI, AI- 08HL, AO-08HL, DM-08TS, DM-16TS.

**[10]** Ascon Firmware Function Block Library.

**[11]** IEC 61131-3 Function Block Library.

**[12]** I/O Function Block Library.