System Manual
PLCcore-iMX35 CODESYS

User Manual for CODESYS-User

Version 1

Edition May 2016

Document No.: L1573e_1

SYS TEC electronic GmbH Am Windrad 2 D-08468 Heinsdorfergrund
Phone: +49 (3765) 38600-0 Fax: +49 (3765) 38600-4100
Web: http://www.systec-electronic.com Mail: info@systec-electronic.com

Status/Changes

Status: released

Date/Version	Section	Changes	Editor
2016/05/18 Version 1	All	Creation	T. Volckmann

This manual includes descriptions for copyrighted products that are not explicitly indicated as such. The absence of the trademark (©) symbol does not infer that a product is not protected. Additionally, xed patents and trademarks are similarly not expressly indicated in this manual.

The information in this document has been carefully checked and is believed to be entirely reliable. However, SYS TEC electronic GmbH assumes no responsibility for any inaccuracies. SYS TEC electronic GmbH neither guarantees nor accepts any liability whatsoever for consequential damages resulting from the use of this manual or its associated product. SYS TEC electronic GmbH reserves the right to alter the information contained herein without prior notification and does not accept responsibility for any damages which might result.

Additionally, SYS TEC electronic GmbH neither guarantees nor assumes any liability for damages arising from the improper usage or improper installation of the hardware or software. SYS TEC electronic GmbH further reserves the right to alter the layout and/or design of the hardware without prior notification and accepts no liability for doing so.

Inform yourselves:

Contact	**Direct**	**Your local distributor**
Address:	SYS TEC electronic GmbH Am Windrad 2 D-08468 Heinsdorfergrund GERMANY	Please find a list of our distributors under: http://www.systec-electronic.com/distributors
Ordering Information:	+49 (0) 37 65 / 38 600-0 info@systec-electronic.com	
Technical Support:	+49 (0) 37 65 / 38 600-0 support@systec-electronic.com	
Fax:	+49 (0) 37 65 / 38 600-4100	
Web Site:	http://www.systec-electronic.com	

1st Edition May 2016

Table of Contents

1 Introduction

Thank you that you have decided for the SYS TEC PLCcore-iMX35. This product provides to you an innovative and high-capacity PLC-kernel. Due to its integrated Target Visualization, high performance as well as extensive on-board periphery, it is particularly suitable for communication and control units for HMI applications.

Please take some time to read through this manual carefully. It contains important information about the commissioning, configuration and programming of the PLCcore-iMX35. It will assist you in getting familiar with the functional range and usage of the PLCcore-iMX35. This document is complemented by other manuals from SYS TEC electronic GmbH and the *CODESYS* programming system of the company 3S Smart Software Solutions GmbH. Table 3 in section 4.1 shows a listing of relevant manuals for the PLCcore-iMX35. Please also refer to those complementary documents.

For more information, optional products, updates et cetera, we recommend you to visit our website: http://www.systec-electronic.com. The content of this website is updated periodically and provides to you downloads of the latest software releases and manual versions.

Declaration of Electro Magnetic Conformity for PLCcore-iMX35 (EMC law)

The PLCcore-iMX35 has been designed to be used as vendor part for the integration into devices (further industrial processing) or as Development Board for laboratory development (hard- and software development).

After the integration into a device or when changes/extensions are made to this product, the conformity to EMC-law again must be assessed and certified. Only thereafter products may be launched onto the market.

The CE-conformity is only valid for the application area described in this document and only under compliance with the following commissioning instructions! The PLCcore-iMX35 is ESD-sensitive and may only be unpacked, used and operated by trained personal at ESD-conform work stations.

The PLCcore-iMX35 is a module for the application in automation technology. It features IEC 61131-3 programmability, uses standard CAN-bus and Ethernet network interfaces and a standardized network protocol. Consequently, development times are short and hardware costs are reasonable. PLC-functionality is created on-board through a CANopen network layer. Hence, it is not necessary for the user to create firmware.

2 Overview / Where to find what?

The PLCcore-iMX35 is based on SYS TEC ECUcore-iMX35 hardware and is extended by PLC-specific functionality (PLC firmware, target visualization). There are different hardware manuals for all hardware components such as the ECUcore-iMX35 and the PLCcore-iMX35 (the hardware of both modules is identical), development boards and reference circuitry. Software-sided, the PLCcore-iMX35 is programmed with IEC 61131-3-conform *CODESYS* programming environment. There are additional manuals for *CODESYS* that describe the handling of programming tools. Those can be found online at http://www.3S-software.com. Table 1 lists up all relevant manuals for the PLCcore-iMX35.

Table 1: Overview of relevant manuals for the PLCcore-iMX35

Information about…	**In which manual?**
Basic information about the PLCcore-iMX35 (configuration, administration, process image, connection assignment, firmware update, reference designs et cetera)	In this manual
Development of user-specific C/C++ applications for the ECUcore-iMX35 / PLCcore-iMX35, VMware-Image of the Linux development system	System Manual ECUcore-iMX35 (Manual no.: L-1569)
Hardware description about the ECUcore-iMX35 / PLCcore-iMX35, reference designs et cetera	Hardware Manual ECUcore-iMX35 (Manual no.: L-1570)
Development Board for the ECUcore-iMX35 / PLCcore-iMX35, reference designs et cetera	Hardware Manual Development Board iMX35 (Manual no.: L-1571)
Complete description of the CODESYS IEC 61131 programming system	Online help about the CODESYS program system, see http://www.3S-software.com

Section 4 of this manual explains the **commissioning of the PLCcore-iMX35** based on the Development Kit for the PLCcore-iMX35.

Section 5 describes the **connection assignment** of the PLCcore-iMX35.

Section 6 explains details about the **application of the PLCcore-iMX35**, e.g. the **setup of the process image**, the **meaning of control elements** and it provides basic information about programming the module. Moreover, information is given about the usage of CAN interfaces in connection with **CANopen**.

Section 7 describes **details about the configuration of the PLCcore-iMX35**, e.g. the configuration of Ethernet and CAN interfaces as well as the Linux Autostart procedure. Furthermore, the **administration of the PLCcore-iMX35** is explained, e.g. the login to the system, the user administration and the execution of software updates.

3 Product Description

The PLCcore-iMX35 as another innovative product extends the SYS TEC electronic GmbH product range within the field of control applications. In the form of an insert-ready core module, it provides to the user a complete and compact PLC. Due to CAN and Ethernet interfaces, the PLCcore-iMX35 is best suitable to realize custom specific HMI (**H**uman **M**achine **I**nterface) applications.

Figure 1: Top view of the PLCcore-iMX35

These are some significant features of the PLCcore-iMX35:

- High-performance CPU kernel (ARM 32-Bit ARM1136JF-S, 532 MHz CPU Clock, 740 MIPS)
- 128 MByte SDRAM Memory, 128 MByte FLASH Memory
- LCD Controller supports up to 800x600 pixel resolution with 24-bit color depth
- Support for Scrollwheel and 4x4 Matrix keypad
- 1x 10/100 Mbps Ethernet LAN interface (with on–board PHY)
- 2x CAN 2.0B interface, usable as CANopen Manager (CiA 302-conform)
- 3x asynchronous serial ports (UART)
- 16 digital inputs, 10 digital outputs
- Externally usable SPI and I^2C
- On-board peripherals: RTC, watchdog, power-fail input
- On-board software: Linux, PLC firmware, CANopen Master, HTTP and FTP server
- Programmable in IEC 61131-3 and in C/C++
- Support of typical PLC control elements (e.g. Run/Stop Switch, Run-LED, Error-LED)
- Linux-based (other user programs may run in parallel)
- Remote Login via Telnet
- Small dimension (78 x 54 mm)

Making PLC available as an insert-ready core module with small dimensions reduces effort and costs significantly for the development of user-specific controls. The PLCcore-iMX35 is also very well suitable as basic component for custom specific HMI devices as well as an intelligent network node for decentralized processing of process signals (CANopen and UDP).

The on-board firmware of the PLCcore-iMX35 contains the entire Target Visualization as well as the PLC runtime environment including CANopen connection with CANopen master functionality. Thus, the module is able to perform human-machine-communication as well as control tasks such as linking in- and outputs or converting rule algorithms. Data and occurrences can be exchanged with other nodes (e.g. superior main controller, I/O slaves and so forth) via CANopen network, Ethernet (UDP

protocol) and serial interfaces (UART). Moreover, the number of in- and outputs either is locally extendable or decentralized via CANopen devices. For this purpose, the CANopen-Chip is suitable. It has also been designed as insert-ready core module for the appliance in user-specific applications.

The PLCcore-iMX35 provides 16 digital inputs (DI0...DI15, 3.3V level), 10 digital outputs (DO0...DO9, 3.3V level) as well as Scrollwheel support. Saving the PLC program in the on-board Flash-Disk of the module allows an automatic restart in case of power breakdown.

Programming the PLCcore-iMX35 takes place according to IEC 61131-3 using the *CODESYS* programming system of the company 3S Smart Software Solutions GmbH (http://www.3S-software.com). Hence, it is possible to program the PLCcore-iMX35 graphically in LD/FBD, SFC and CFC or textually in IL or ST. Downloading the PLC program onto the module takes place via Ethernet. Addressing in- and outputs and creating a process image follows the SYS TEC scheme for compact control units. Like all other SYS TEC controls, the PLCcore-iMX35 supports backward documentation of the PLC program as well as the debug functionality including watching and setting variables, single cycles, breakpoints and single steps.

The integrated Target Visualization is based on the visualization component of the CODESYS programming system of the company 3S Smart Software Solutions GmbH (http://www.3S-software.com). It enables for displaying of process values from the PLC as well as forwarding of operator actions to the PLC (e.g. entries via Touchscreen and Srollwheel).

The PLCcore-iMX35 is based on Embedded Linux as operating system. This allows for an execution of other user-specific programs while PLC firmware is running.

The Embedded Linux applied to the PLCcore-iMX35 is licensed under GNU General Public License, version 2. Appendix B contains the license text. All sources of LinuxBSP are included in the software package **SO-1121** ("VMware-Image of the Linux development system for the ECUcore-iMX35"). If you require the LinuxBSP sources independently from the VMware-Image of the Linux development system, please contact our support:

support@systec-electronic.com

The PLC system and the PLC- and C/C++ programs developed by the user are **not** subject to GNU General Public License!

4 Development Kit PLCcore-iMX35

4.1 Overview

The Development Kit PLCcore-iMX35 is a high-capacity, complete package at a particularly favorable price. Based on a compact PLC with integrated target visualization, it enables the user to develop own, custom specific HMI devices.

Figure 2: Development Kit PLCcore-iMX35

The Development Kit PLCcore-iMX35 ensures quick and problem-free commissioning of the PLCcore-iMX35. Therefore, it combines all hard- and software components that are necessary to create own HMI applications: the core module PLCcore-iMX35, the corresponding Development Board containing a QVGA LCD Display, I/O periphery and numerous interfaces, the IEC 61131 programming system CODESYS as well as further accessory. Thus, the Development Kit forms the ideal platform for developing user-specific HMI applications based on the PLCcore-iMX35. It allows for a cost-efficient introduction into the world of decentralized automation technology. All components included in the Kit enable in- and output extensions of the PLCcore-iMX35 through CANopen-I/O-assemblies. Thus, the Development Kit may also be used for projects that require PLC with network connection.

The Development Kit PLCcore-iMX35 contains the following hardware components:

- PLCcore-iMX35
- Development Board for the PLCcore-iMX35, incl.:
  - 320x240 pixel QVGA LCD Display
  - Scrollwheel (on-board)
  - 12V – 1,5A DC Power adapter
- Ethernet cable
- RS232 cable
- RS485 connector
- CD with programming software, examples, documentation and other tools

The Development Board included in the Kit facilitates quick commissioning of the PLCcore-iMX35 and simplifies the design of prototypes for user-specific HMI applications based on this module. Among other equipment, the Development Board comprises different power supply possibilities, a 320x240 pixel QVGA LCD Display, Ethernet interface, 2 CAN interfaces, 4 push buttons and 4 LED as control elements for digital in- and outputs and it comprises a Scrollwheel and a connector for a 4x4 Matrix

Keypad. Signals that are available from plug connectors of the PLCcore-iMX35 are linked to pin header connectors and enable easy connection of own peripheral circuitry. Hence, the Development Board forms an ideal experimentation and testing platform for the PLCcore-iMX35.

The CODESYS programming system included in the Kit serves as software development platform and as debug environment for the PLCcore-iMX35. Thus, the module can either be programmed graphically in KOP/FUB, AS and CFC or textually in IL or ST. Downloading the PLC program onto the module takes place via Ethernet. High-capacity debug functionality such as watching and setting variables, single cycles, breakpoints and single steps simplify the development and commissioning of user software for this module.

4.2 Electric commissioning of the Development Kit PLCcore-iMX35

A power adapter necessary for running the Development Kit PLCcore-iMX35 as well as Ethernet and RS232 cables are already included in the Kit delivery. For commissioning the Kit, it is essential to use at least the power supply connections (X100/X101), COM0 (X701A) and ETH0 (X702). Furthermore, connection CAN0 (X801A) is recommended. Table 2 provides an overview over the connections of the Development Kit PLCcore-iMX35.

Table 2: Connections of the Development Kit PLCcore-iMX35

Connection	Labeling on the Development Board	Remark
Power supply	X100 or X101	The power adapter included in the delivery is intended for direct connection to X101.
ETH0 (Ethernet)	X702	This interface serves as communication interface with the Programming PC and is necessary for the program download, besides can be used freely for the user program.
COM0 (RS232)	X701A	This interface is used for the configuration of the unit (e.g. setting the IP-address) and can be used freely for general operation of the user program.
COM1 (RS232)	X701B	Interface can be used freely for the user program.
COM2 (RS485)	X700	Interface can be used freely for the user program.
CAN0 (CAN)	X801A	Interface can be used freely for the user program.
CAN1 (CAN)	X801B	Interface can be used freely for the user program.

Figure 3 shows the positioning of the most important connections of the Development Board for the PLCcore-iMX35. Instead of using the power adapter included in the Kit, the power supply may optionally take place via X100 with an external source of 12V/1,5A.

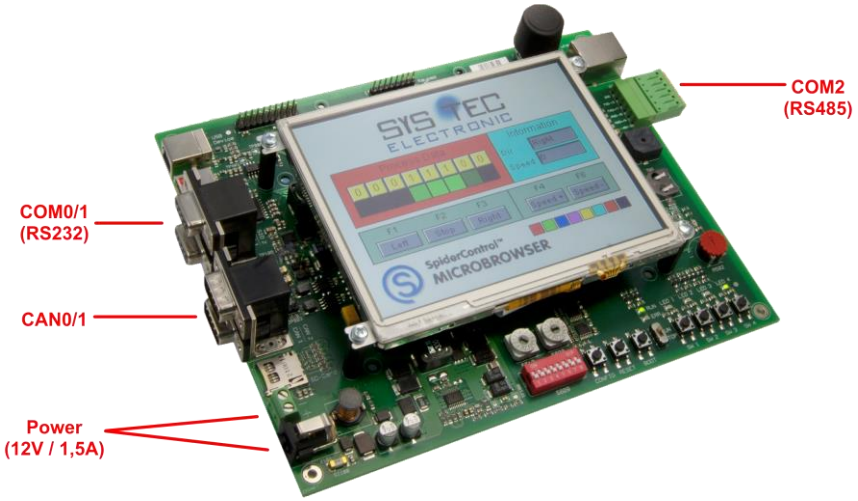


Figure 3: Positioning of most important connections on the Development Board for the PLCcore-iMX35

Advice: **Upon commissioning, cables for Ethernet (ETH0, X702) and RS232 (COM0, X701A) must be connected prior to activating the power supply (X100 / X101).**

4.3 Control elements of the Development Kit PLCcore-iMX35

The Development Kit PLCcore-iMX35 allows for easy commissioning of the PLCcore-iMX35. It has available various control elements to configure the module and to simulate in- and outputs for the usage of the PLCcore-iMX35 as PLC kernel. In Table 3 control elements of the Development Board are listed and their meaning is described.

Table 3: Control elements of the Development Board for the PLCcore-iMX35

Control element	Name	Meaning
Pushbutton 0	S604	Digital Input DI0 (Process Image: %IX0.0)
Pushbutton 1	S605	Digital Input DI1 (Process Image: %IX0.1)
Pushbutton 2	S606	Digital Input DI2 (Process Image: %IX0.2)
Pushbutton 3	S607	Digital Input DI3 (Process Image: %IX0.3)
LED 0	D602	Digital Output DO0 (Process Image: %QX0.0)
LED 1	D603	Digital Output DO1 (Process Image: %QX0.1)
LED 2	D604	Digital Output DO2 (Process Image: %QX0.2)
LED 3	D605	Digital Output DO3 (Process Image: %QX0.3)
Run/Stop Switch	S603	Run / Stop to operate the PLC program
Run-LED	D600	Display of activity state of the PLC
Error-LED	D601	Display of error state of the PLC
Hex-Encoding Switch	S608/S610	Configuration of node address CAN0
DIP-Switch	S609	Configuration of bitrate and master mode CAN0

Table 6 in section 6.4.1 provides a complete listing of the process image.

4.4 Optional accessory

4.4.1 USB-RS232 Adapter Cable

The SYS TEC USB-RS232 Adapter Cable (order number 3234000) provides a RS232 interface via an USB-Port of the PC. Together with a terminal program, it enables the configuration of the PLCcore-iMX35 from PCs, e.g. laptop computers which do not have RS232 interfaces any more (see section 6.1).

Figure 4: SYS TEC USB-RS232 Adapter Cable

5 Pinout of the PLCcore-iMX35

Connections of the PLCcore-iMX35 are directed to the outside via two female headers that are double-row and mounted on the bottom of the module (X600A/B, see Figure 5). Appropriate pin header connectors as correspondent to the PLCcore-iMX35 are available from company “W + P“:

W+P name: SMT Pin Headers, 1.27mm Pitch, Vertical, Double Row - 1.0mm Body
W+P order number: 7072-100-10-00-10-PPST (deliverable in other sizes)

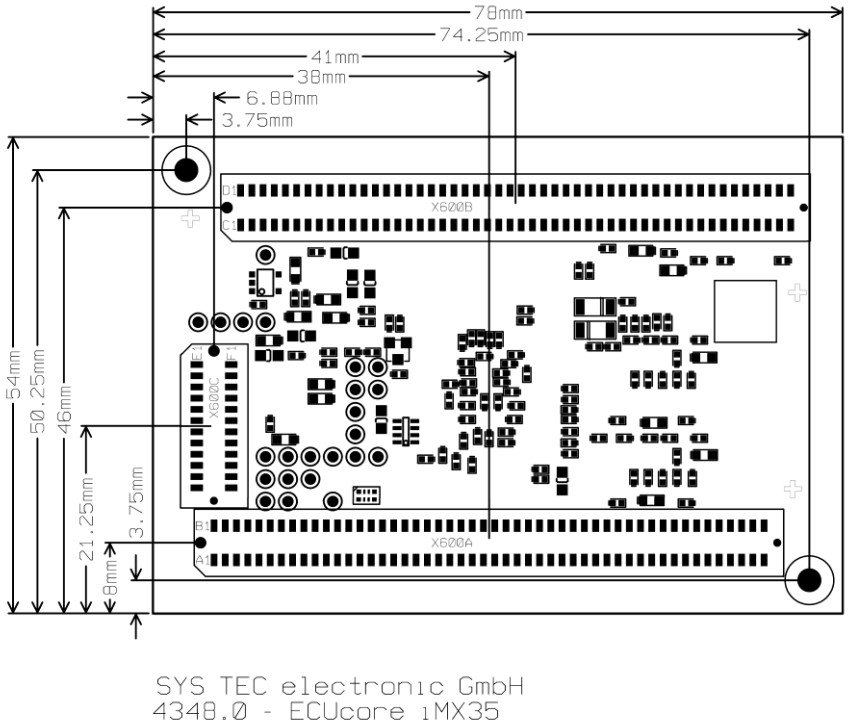


Figure 5: Pinout of the PLCcore-iMX35 - top view

Figure 5 exemplifies the positioning of female headers (X600A/B) on the PLCcore-iMX35. The complete connection assignment of this module is listed up in Table 4. The additional female header X600C shown in Figure 5 is reserved for a JTAG interface. It is only equipped on special development boards. For the usage of the PLCcore-iMX35 as PLC kernel it is without any importance. A detailed description of all module connectors is located in the Hardware Manual ECUcore-iMX35 (Manual no.: L-1570). Appendix A includes reference designs for using the PLCcore-iMX35 in customer-specific applications.

Table 4: Connections of the PLCcore-iMX35, completely, sorted by connection pin

Signal	Pin	Pin	Signal		Signal	Pin	Pin	Signal
GND	A01	B01	GND		GND	C01	D01	2V5_EPHY
/BOOT	A02	B02	/MR		Eth_Tx-	C02	D02	GND
/BOOTSTRAP_1	A03	B03	/RESET_IN		Eth_Tx+	C03	D03	Speed
VSTBY	A04	B04	/PFI		Eth_Rx+	C04	D04	Link/Act
/BOOTSTRAP_0	A05	B05	WDI		Eth_Rx-	C05	D05	GND
GND	A06	B06	/PFO		GND	C06	D06	GPIO1_6
RXD1	A07	B07	GND		GPIO1_0	C07	D07	GPIO1_5

Signal	Pin	Pin	Signal		Signal	Pin	Pin	Signal
TXD1	A08	B08	RTS2		GPIO1_1	C08	D08	GPIO1_4
RTS1	A09	B09	CTS2		GND	C09	D09	GND
CTS1	A10	B10	RTS3		SD2_DATA0	C10	D10	GPIO1_3
GPIO2_12	A11	B11	CTS3		SD2_DATA1	C11	D11	USBOTG_OC
GND	A12	B12	GND		SD2_DATA2	C12	D12	USBOTG_PWR
TXD2	A13	B13	TXD3		SD2_DATA3	C13	D13	USBPHY1_VBUS
RXD2	A14	B14	RXD3		SD2_CLK	C14	D14	SD2_CMD
NVCC_3V3	A15	B15	GPIO1_26		GND	C15	D15	GPIO1_31
NVCC_3V3	A16	B16	GPIO2_18		GPIO1_28	C16	D16	GND
GND	A17	B17	Unused		Unused	C17	D17	CAPTURE
USBPHY2_DP	A18	B18	Unused		GPIO2_12	C18	D18	GPIO1_25
USBPHY2_DM	A19	B19	GND		GPIO2_13	C19	D19	COMPARE
USBPHY1_UID	A20	B20	USBPHY1_DP		GPIO2_14	C20	D20	CLKO
Unused	A21	B21	USBPHY1_DM		GPIO2_15	C21	D21	GPIO2_26
GND	A22	B22	GND		GPIO2_17	C22	D22	GPIO2_28
I2C2_DAT	A23	B23	CAN1_TX		GPIO2_25	C23	D23	GND
I2C2_CLK	A24	B24	CAN1_RX		GND	C24	D24	BACKL_EN
GND	A25	B25	GPIO1_24		GPIO1_2	C25	D25	GPIO2_31
GPIO3_25	A26	B26	GND		LCD_CONTRAST	C26	D26	GPIO2_29
GND3_26	A27	B27	/RESET		GPIO2_30	C27	D27	GPIO2_19
Unused	A28	B28	/PORESET		GPIO2_20	C28	D28	GND
/EN_IO3V3	A29	B29	Unused		GND	C29	D29	GPIO2_21
Unused	A30	B30	CAN2_RX		GPIO2_22	C30	D30	Unused
GND	A31	B31	CAN2_TX		LCD_TXout0+	C31	D31	LCD_TXout0-
CSPI1_SS0	A32	B32	GND		LCD_TXout1+	C32	D32	LCD_TXout1-
CSPI1_SS2/PWMO	A33	B33	CSPI1_SS1		LCD_TXout2+	C33	D33	GND
CSPI1_MOSI	A34	B34	CSPI1_MISO		LCD_TXout2-	C34	D34	LCD_TXoutCLK+
CSPI1_SS3	A35	B35	CSPI1_SCLK		GND	C35	D35	LCD_TXoutCLK-
Unused	A36	B36	Unused		LCD_R0	C36	D36	LCD_R1
GND	A37	B37	SD1_CLK		LCD_R2	C37	D37	LCD_R3
SD1_CMD	A38	B38	GND		LCD_R4	C38	D38	LCD_R5
SD1_DATA0	A39	B39	SD1_DATA1		LCD_G0	C39	D39	GND
SD1_DATA2	A40	B40	SD1_DATA3		LCD_G1	C40	D40	LCD_G2
MATRIX_C1	A41	B41	MATRIX_C0		GND	C41	D41	LCD_G3
MATRIX_C3	A42	B42	MATRIX_C2		LCD_G4	C42	D42	LCD_G5
GND	A43	B43	MATRIX_R0		LCD_B0	C43	D43	LCD_B1
MATRIX_R1	A44	B44	GND		LCD_B2	C44	D44	LCD_B3
MATRIX_R3	A45	B45	MATRIX_R2		LCD_B4	C45	D45	GND
GPIO1_8	A46	B46	GPIO1_9		LCD_B5	C46	D46	/LVDS_PWD
GPIO1_12	A47	B47	GPIO1_13		GND	C47	D47	GPIO1_15
VBAT	A48	B48	GPIO1_14		LCD_DEN	C48	D48	LCD_DCLK
GND	A49	B49	GND		LCD_HSYNC	C49	D49	LCD_VSYNC
+3V3	A50	B50	+3V3		GND	C50	D50	GND

Table 5 is a subset of Table 4 and only includes all in- and outputs of the PLCcore-iMX35 sorted by their function.

Table 5: Connections of the PLCcore-iMX35, only I/O, sorted by function

Connector	I/O-Pin	PLC Function 1	PLC Function 2 A=alternative, S=simultaneous
D27	GPIO2_19	DI0 [Switch0]	
C28	GPIO2_20	DI1 [Switch1]	
D29	GPIO2_21	DI2 [Switch2]	
C30	GPIO2_22	DI3 [Switch3]	
B25	GPIO1_24	DI4	
D18	GPIO1_25	DI5	
B15	GPIO1_26	DI6	
D15	GPIO1_31	DI7	
D14	GPIO2_0 (SD2_CMD)	DI8	
A47	GPIO1_12	DI9	
B16	GPIO2_18	DI10	
B9	GPIO3_13	DI11	
A26	GPIO3_25	DI12	
A27	GPIO3_26	DI13	
D21	GPIO2_26	DI14	
C25	GPIO1_2	DI15	
D22	GPIO2_28	DO0 [LED0]	
D26	GPIO2_29	DO1 [LED1]	
C27	GPIO2_30	DO2 [LED2]	
D25	GPIO2_31	DO3 [LED3]	
C7	GPIO1_0	DO4	
C8	GPIO1_1	DO5	
D10	GPIO1_3	DO6	
D6	GPIO1_6	DO7	
C14	GPIO2_1 (SD2_CLK)	DO8	
C21	GPIO2_15	DO9	
B41	MATRIX_C0	MATRIX_C0	
A41	MATRIX_C1	MATRIX_C1	
B42	MATRIX_C2	MATRIX_C2	
A42	MATRIX_C3	MATRIX_C3	
B43	MATRIX_R0	MATRIX_R0	
A44	MATRIX_R1	MATRIX_R1	
B45	MATRIX_R2	MATRIX_R2	
A45	MATRIX_R3	MATRIX_R3	
D17	CAPTURE	Scrollwheel DIR	
D19	COMPARE	Scrollwheel CLK	
C22	GPIO2_17	Scrollwheel Button	
A46	GPIO1_8	Error-LED	
B46	GPIO1_9	Run-LED	
B8	GPIO3_12 (RTS2)	R/S-Switch (High: "Run")	

If no Run/Stop Switch is intended for the usage of the PLCcore-iMX35 on an application-specific baseboard, the coding for *"Run"* must be hard-wired at the module connections (also see reference design in Appendix A).

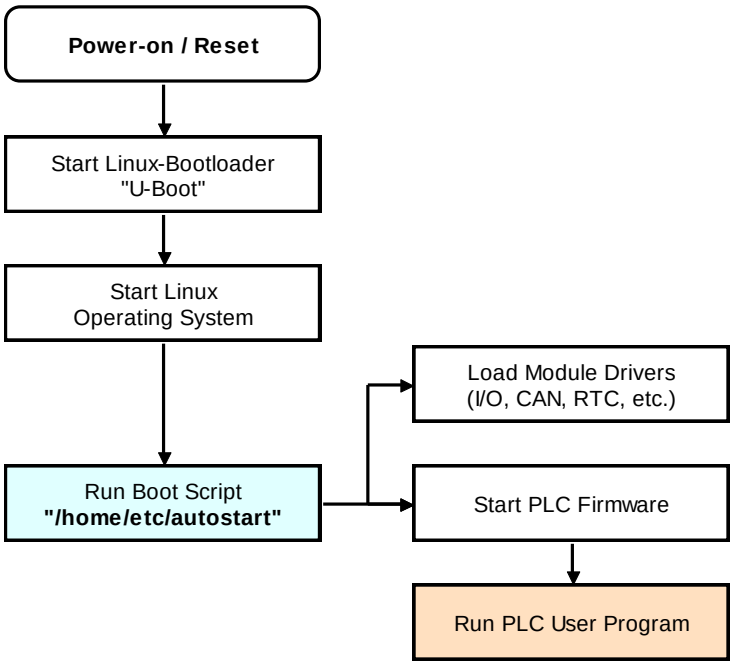
6 PLC Functionality of the PLCcore-iMX35

6.1 Overview

The PLCcore-iMX35 realizes a complete Linux-based compact PLC as an insert-ready core ("Core"). There, the PLCcore-iMX35 is based on the hardware ECUcore-iMX35 and extends it by PLC-specific functionality (PLC firmware, Target Visualization). Both modules, the ECUcore-iMX35 and the PLCcore-iMX35, use the same Embedded Linux as operating system. Consequently, the configuration and the C/C++ programming of the PLCcore-iMX35 are almost identical with the ECUcore-iMX35.

6.2 System start of the PLCcore-iMX35

By default, the PLCcore-iMX35 loads all necessary firmware components upon Power-on or Reset and starts running the PLC program afterwards. Hence, the PLCcore-iMX35 is suitable for the usage in autarchic control systems. In case of power breakdown, such systems resume the execution of the PLC program independently and without user intervention. Figure 6 shows the system start in detail:



For more details on how to deactivate the autarchic Linux start and to activate the “U-Boot" command prompt compare section 7.2.

Details about the start script ***"/home/etc/autostart"*** are covered in section 7.4.

For detailed information about PLC programming of the PLCcore-iMX35 compare section 6.3.

Figure 6: System start of the PLCcore-iMX35

6.3 Programming the PLCcore-iMX35

The PLCcore-iMX35 is programmed with IEC 61131-3-conform *CODESYS* programming environment. There exist additional manuals about *CODESYS* that describe the handling of this programming tool. Those are online available at 3S Smart Software Solutions GmbH, see http://www.3S-Software.com. All manuals related to the PLCcore-iMX35 are listed in Table 1.

PLCcore-iMX35 firmware is based on standard firmware for SYS TEC`s compact control units. Consequently, it shows identical properties like other SYS TEC control systems. This affects especially the process image setup (see section 6.4) as well as the functionality of control elements (Hex-Encoding switch, DIP-Switch, Run/Stop Switch, Run-LED, Error-LED).

Detailed information about using the CAN interfaces in connection with CANopen is provided in section 6.6.

6.4 Process image of the PLCcore-iMX35

6.4.1 Local In- and Outputs

Compared to other SYS TEC compact control systems, the process image depends on the device tree structure of the user-specific CODESYS project. All in- and outputs listed in Table 6 are supported by the PLCcore-iMX35.

Table 6: Assignment of in- and outputs to the process image of the PLCcore-iMX35

I/O of the PLCcore-iMX35	Address and Data type in the Process Image	
DI0 ... DI7	**%IB0.0** %IX0.0 … %IX0.7	as Byte with DI0 … DI7 as single Bit for each input
DI8 ... DI15	**%IB1.0** %IX1.0 … %IX1.7	as Byte with DI8 … DI15 as single Bit for each input
AIN0	**%IW8.0**	15Bit + sign(0 … + 32767)
AIN1	**%IW10.0**	15Bit + sign(0 … + 32767)
AIN2	**%IW12.0**	15Bit + sign(0 … + 32767)
AIN3	**%IW14.0**	15Bit + sign(0 … + 32767)
On-board Temperature Sensor	**%ID72.0**	31Bit + sign as 1/10000 °C
DO0 ... DO7	**%QB0.0** %QX0.0 … %QX0.7	as Byte with DO0 … DO7 as single Bit for each output
DO8 … DO9	**%QB1.0** %QX1.0	as Byte with DO8 … DO9 as single Bit for each output

In- and outputs of the PLCcore-iMX35 are not negated in the process image. Hence, the H-level at one input leads to value "1" at the corresponding address in the process image. Contrariwise, value "1" in the process image leads to an H-level at the appropriate output.

6.5 Communication interfaces

6.5.1 Serial interfaces

The PLCcore-iMX35 features 3 serial interfaces (COM0 … COM2). COM0 and COM1 are used for RS-232 mode while COM2 can be used as RS-485 interface. Details about hardware activation are included in the *"Hardware Manual Development Board ECUcore-iMX35"* (Manual no.: L-1571).

COM0: Interface COM0 primarily serves as service interface to administer the PLCcore-iMX35. By default, in boot script *"/etc/inittab"* it is assigned to the Linux process *"getty"* and is used as Linux console to administer the PLCcore-iMX35.

COM1/2: Interface COM1 is disposable and support data exchange between the PLCcore-iMX35 and other field devices kept under control of the PLC program.

6.5.2 CAN interfaces

The PLCcore-iMX35 features 2 CAN interfaces (CAN0 … CAN1). Details about the hardware activation are included in the *"Hardware Manual Development Board ECUcore-iMX35"* (Manual no.: L-1571).

The CAN interfaces allow for data exchange with other devices via network variables.

Section 6.6 provides detailed information about the usage of the CAN interfaces in connection with CANopen.

6.5.3 Ethernet interfaces

The PLCcore-iMX35 features 1 Ethernet interface (ETH0). Details about the hardware activation are included in the *"Hardware Manual Development Board ECUcore-iMX35"* (Manual no.: L-1571).

The Ethernet interface serves as service interface to administer the PLCcore-iMX35 and it enables downloading, debugging and monitoring of CODESYS-applications from the PC.

6.6 Using CANopen for CAN interfaces

The PLCcore-iMX35 features 2 CAN interfaces (CAN0 … CAN1), both are usable as CANopen Manager (conform to CiA Draft Standard 302). The configuration of this interface (active/inactive, node number, Bitrate, Master on/off) is described in section 6.6.1.

The CAN interface allow for data exchange with other devices via network variables. Therefore the CAN-Bus, CODESYS CANopen-Manager and the corresponding CANopen devices must be added to the CODESYS project device tree (see section 6.6.1).

The CANopen services **PDO** (***P***rocess ***D***ata ***O***bjects) and **SDO** (***S***ervice ***D***ata ***O***bjects) are two separate mechanisms for data exchange between single field bus devices. Process data sent from a node (**PDO**) are available as broadcast to interested receivers. PDOs are limited to 1 CAN telegram and there with to 8 Byte user data maximum because PDOs are executed as non-receipt broadcast messages. On the contrary, **SDO** transfers are based on logical point-to-point connections ("Peer to Peer") between two nodes and allow the receipted exchange of data packages that may be larger than 8 Bytes. Those data packages are transferred internally via an appropriate amount of CAN telegrams. Both services are applicable for interface CAN0 as well as for CAN1 of the PLCcore-iMX35.

Network variables should mainly be used for PDO-based data exchange. Network variables represent the easiest way of data exchange with other CANopen nodes. Accessing network variables within a PLC program takes place in the same way as accessing internal, local variables of the PLC. Hence, for PLC programmers it is not of importance if e.g. an input variable is allocated to a local input of the control or if it represents the input of a decentralized extension module. The application of network variables is based on the integration of DCF files that are generated by an appropriate CANopen configurator. On the one hand, DCF files describe communication parameters of any device (CAN

Identifier, etc.) and on the other hand, they allocate network variables to the Bytes of a CAN telegram (mapping). The application of network variables only requires basic knowledge about CANopen.

In a CANopen network, exchanging PDOs only takes place in status *"OPERATIONAL"*. If the PLCcore-iMX35 is not in this status, it does not process PDOs (neither for send-site nor for receive-site) and consequently, it does not update the content of network variables. The CANopen Manager is in charge of setting the operational status *"OPERATIONAL"*, *"PRE-OPERATIONAL"* etc. (mostly also called "CANopen Master"). In typical CANopen networks, a programmable node in the form of a PLC is used as CANopen-Manager. The PLCcore-iMX35 is optionally able to take over tasks of the CANopen Manager. How the Manager is activated is described in section 6.6.1.

As CANopen Manager, the PLCcore-iMX35 is able to parameterize the CANopen I/O devices ("CANopen-Slaves") that are connected to the CAN bus. Therefore, upon system start via SDO it transfers DCF files generated by the CANopen configurator to the respective nodes.

6.6.1 CAN0 configuration for CANopen in CODESYS

6.6.1.1 Add CANopen-Device

The following steps are necessary to add a new CANopen device to a CODESYS project:

1. Add the CAN fieldbus to the device tree and configuration of the node-id and baud rate (double clicking the CAN-bus entry in the device tree)
2. Add and configure the CANopen-Manager (double click the CANopen Manager in the device tree)
3. Add the CANopen Slave (e.g. SYSTEC-F40 Module)

The following steps are related to step 3:

1. Install the corresponding EDS file that creates the Devdesc (Menu > Device Repository – Button “Install”)

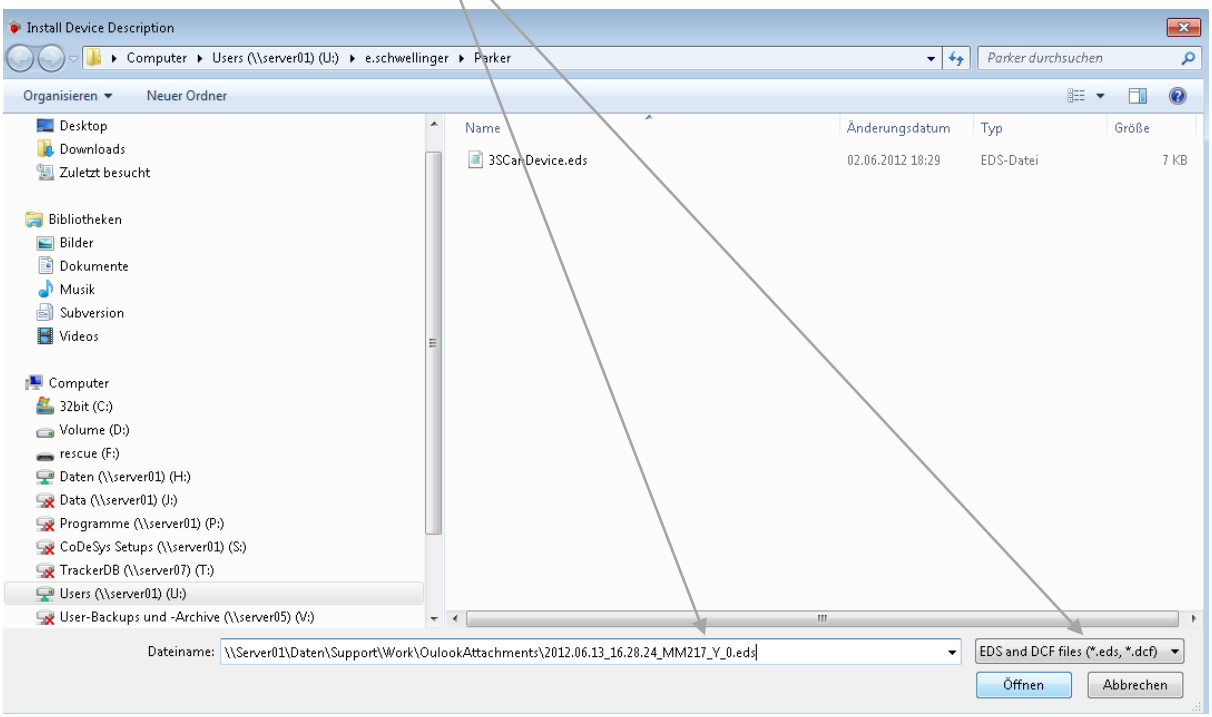


Figure 1: Install the SYSTEC F40 into the device repository

2. Add the slave to the CANopen-Manager (Right click the CANopen Manager > “Add Device”)

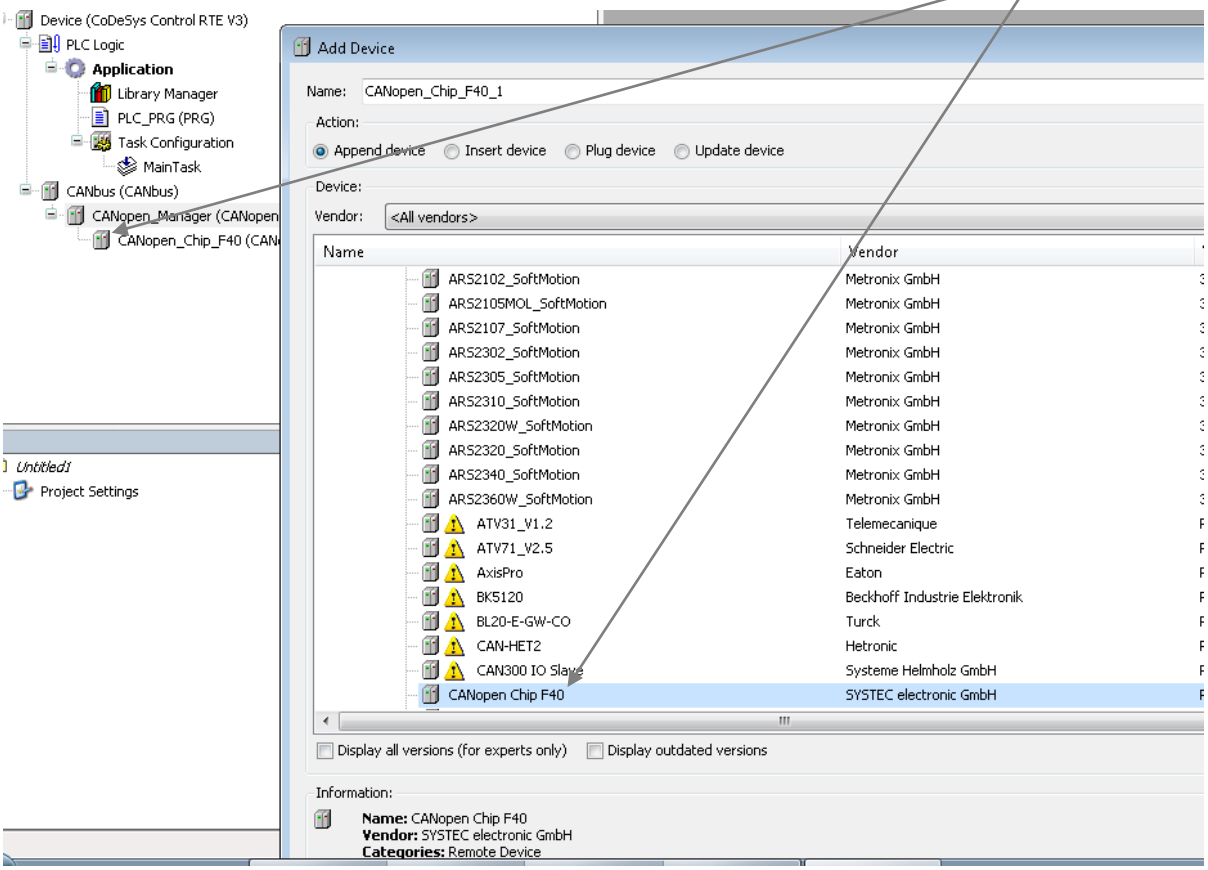


Figure 2: Append the SYSTEC F40 module to the CANopen-Manager

3. The default configuration of the SYSTEC-F40 module is shown in Figure 3:

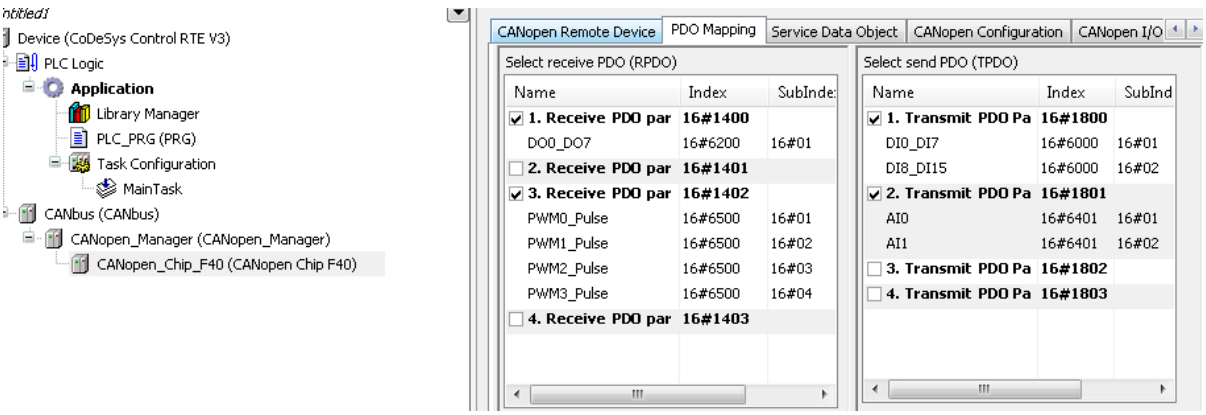


Figure 3: Default configuration of the SYSTEC F40 module

6.6.1.2 Variable-Mapping

The variable mapping is done via “CANopen I/O Mapping” tab. Network variables can be defined as follows:

- Network variables as global, project specific variable – use column “Variable”
- Network variables assigned to an existing variable – use column “Mapping”

CODESYS does not support both methods for the same variable simultaneously.

The mapped I/O entries of the CANopen devices are accessible via the IEC addresses of the process image (%IBnn, %IWnn, … %QBnn, %QWnn etc.). Those variables must be declared as follows to sue them in the SPS application, e.g.:

```
MyVar  AT%IBnn  :  BYTE;
```

“nn” represents the corresponding IEC address of the network variable. The IEC address is related to the position of the CANopen device in the device tree and is automatically assigned by the CODESYS-IDE.

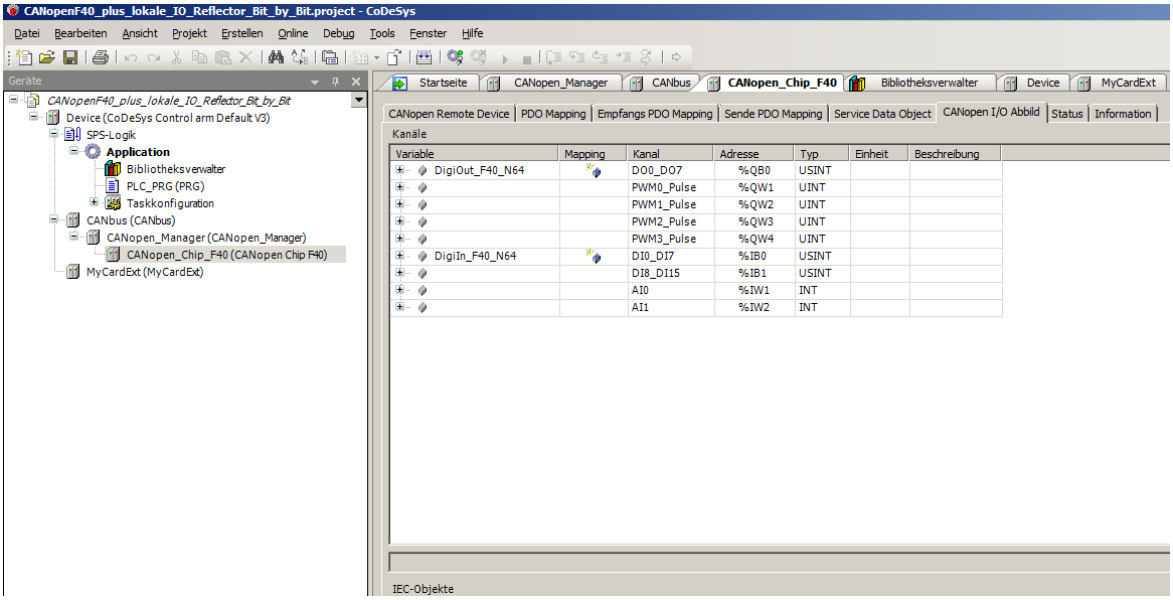


Figure 4: Variable mapping of the F407 module

The first method has been chosen in the shown example in Figure 4.

Therefore all configured CANopen network variables are available in the entire project and be used as follows:

```
DigiOut_F40_N64.0 := DigiIn_F40_N64.0;
DigiOut_F40_N64.1 := DigiIn_F40_N64.1;
DigiOut_F40_N64.2 := DigiIn_F40_N64.2;
DigiOut_F40_N64.3 := DigiIn_F40_N64.3;
```

The bits 0…3 of the input bytes DI0 … DI7 module are assigned to bits 0…3 of the digital output bytes DO0 … DO7 of the F40 module.

6.6.2 Additional CAN-Interfaces

The firmware of the PLCcore-iMX35 is able to handle multiple CAN-Interfaces at the same time like PLCcore-5484 or PLCmodule-C32 does.

Note: Using more than one CAN-Interface is not supported by CODESYS.

More CAN interfaces can be connected to the module if necessary. Please contact our support for further information:

support@systece-electronic.com

6.7 Target Visualization

A “Visualization Manager” must be added to enable visualization and control via touchscreen. All required visualization objects are inserted below the “Visualization Manager” and developed using corresponding tools in CODESYS. A detailed description on how to use the tools for the development of visualization objects in CODESYS is available in the corresponding manuals, online (see http://www.3S-software.com) and using the online-help that is included in the CODESYS programming system.

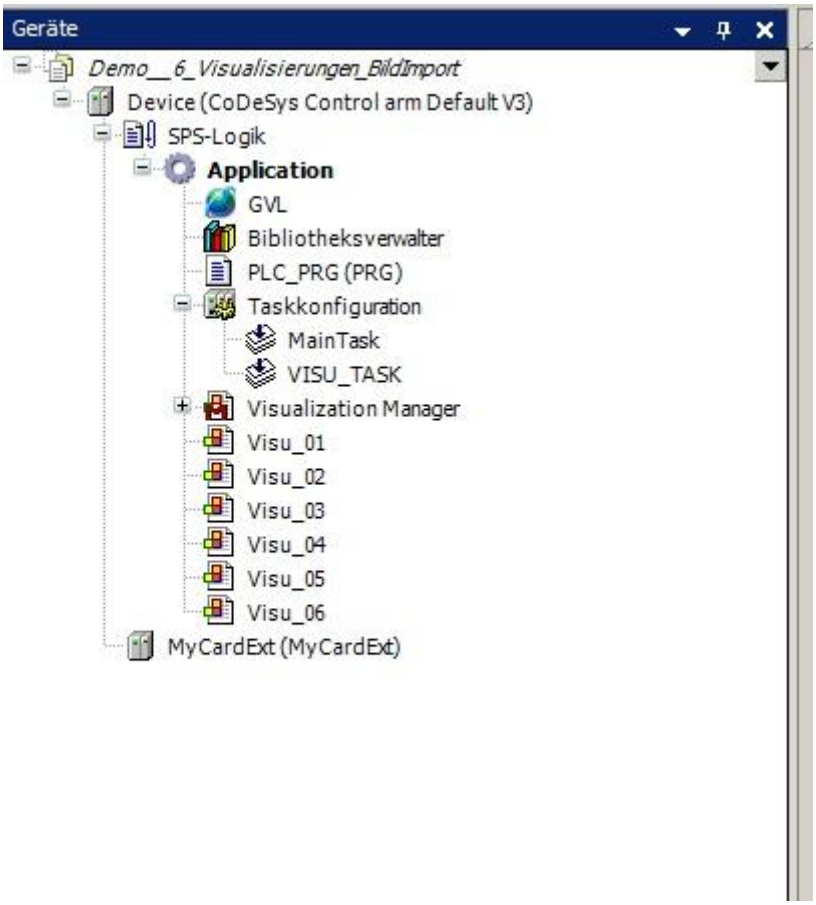


Figure 5: Device tree when using visualization in CODESYS

7 Configuration and Administration of the PLCcore-iMX35

7.1 System requirements and necessary software tools

The administration of the PLCcore-iMX35 requires any Windows or Linux computer that has available an Ethernet interface and a serial interface (RS232). As alternative solution to the on-board serial interface, SYS TEC offers a USB-RS232 Adapter Cable (order number 3234000, see section 4.4.1) that provides an appropriate RS232 interface via USB port.

All examples referred to in this manual are based on an administration of the PLCcore-iMX35 using a Windows computer. Procedures using a Linux computer would be analogous.

To administrate the PLCcore-iMX35 the following software tools are necessary:

Terminal program

A Terminal program allows the communication with the **command shell** of the PLCcore-iMX35 via a **serial RS232 connection to COM0 of the PLCcore-iMX35**. This is required for the Ethernet configuration of the PLCcore-iMX35 as described in section 7.3. After completing the Ethernet configuration, all further commands can either be entered in the Terminal program or alternatively in a Telnet client (see below).

Suitable as Terminal program would be *"HyperTerminal"* which is included in the Windows delivery or *"TeraTerm"* which is available as Open Source and meets higher demands (downloadable from: http://ttssh2.sourceforge.jp).

Telnet client

Telnet-Client allows the communication with **command shell** of the PLCcore-iMX35 via **Ethernet connection to ETH0 of the PLCcore-iMX35**. Using Telnet clients requires a completed Ethernet configuration of the PLCcore-iMX35 according to section 7.3. As alternative solution to Telnet client, all commands can be edited via a Terminal program (to COM0 of the PLCcore-iMX35).

Suitable as Telnet client would be *"Telnet"* which is included in the Windows delivery or *"TeraTerm"* which can also be used as Terminal program (see above).

FTP client

An FTP client allows for file exchange between the PLCcore-iMX35 (ETH0) and the computer. This allows for example **editing configuration files** by transferring those from the PLCcore-iMX35 onto the computer where they can be edited and get transferred back to the PLCcore-iMX35. Downloading files onto the PLCcore-iMX35 is also necessary to **update the PLC firmware**. (Advice: The update of *PLC firmware* is not identical with the update of the *PLC user program*. The PLC program is directly transferred to the module from the *CODESYS* programming environment. No additional software is needed for that.)

Suitable as FTP client would be *"WinSCP"* which is available as Open Source (download from: http://winscp.net). It only consists of one EXE file that needs no installation and can be booted immediately. Furthermore, freeware *"Core FTP LE"* (downloadable from: http://www.coreftp.com) or *"Total Commander"* (integrated in the file manager) are suitable as FTP client.

TFTP server The TFTP server is necessary to update the Linux-Image on the PLCcore-iMX35. Freeware *"TFTPD32"* (download from: http://tftpd32.jounin.net) is suitable as TFTP server. It only consists of one EXE file that needs no installation and can be booted immediately.

For programs that communicate via Ethernet interface, such as FTP client or TFTP server, it must be paid attention to that rights in the Windows-Firewall are released. Usually Firewalls signal when a program seeks access to the network and asks if this access should be permitted or denied. In this case access is to be permitted.

7.2 Activation/Deactivation of Linux Autostart

During standard operation mode, the bootloader "U-Boot" automatically starts the Linux operating system of the module after Reset (or Power-on). Afterwards, the operating system loads all further software components and controls the PLC program execution (see section 6.1). For service purposes, such as configuring the Ethernet interface (see section 7.3) or updating the Linux-Image (see section 7.12.2), it is necessary to disable this Autostart mode and to switch to "U-Boot" command prompt instead (configuration mode).

The automatic boot of Linux operating system is connected with the **simultaneous compliance** with various conditions ("AND relation"). Consequently, for disabling Linux Autostart, it is sufficient to simply **not comply** with one of the conditions.

Table 7 lists up all conditions that are verified by the bootloader “U-Boot”. All of them must be complied with to start an Autostart for the Linux-Image.

Table 7: Conditions for booting Linux

No.	Condition	Remark
1	Connection "/BOOT" = High (pushbutton S602 on the Development Board **not** pressed)	The Linux Autostart is released only if the signal “/BOOT” is at H-level (“/BOOT” is not active). The position of connection "/BOOT" on the module pin connector is defined in the Hardware Manual PLCcore-iMX35 (Manual no.: L-1570).
2	**No** abort of Autostart via COM0 of the PLCcore-iMX35	If the conditions above are met, "U-Boot" checks the serial interface COM0 of the PLCcore-iMX35 for about 1 second after Reset regarding the reception of SPACE signals (ASCII 20H). If such a signal is received within that time, "U-Boot" will disable the Linux Autostart and will activate its own command prompt instead.

According to Table 7, the Linux boot is disabled after Reset (e.g. pushbutton S601 on the Development Board) and the "U-Boot" command prompt is activated instead if the following conditions occur:

(1) /BOOT = "Low" Development Board: "/BOOT" = pushbutton S602

- OR -

(2) Reception of a SPACE signal (ASCII 20H) within 1 second after Reset

After activating the Reset pushbutton (e.g. pushbutton S601 on the Development Board), the "U-Boot" command prompt answers.

Communicating with the bootloader "U-Boot" only takes place via the serial interface COM0 of the PLCcore-iMX35. As receiver on the computer one of the terminal programs must be started (e.g. HyperTerminal or TeraTerm, see section 7.1) and must be configured as follows (see Figure 7):

- 115200 Baud
- 8 Data bit
- 1 Stop bit
- no parity
- no flow control

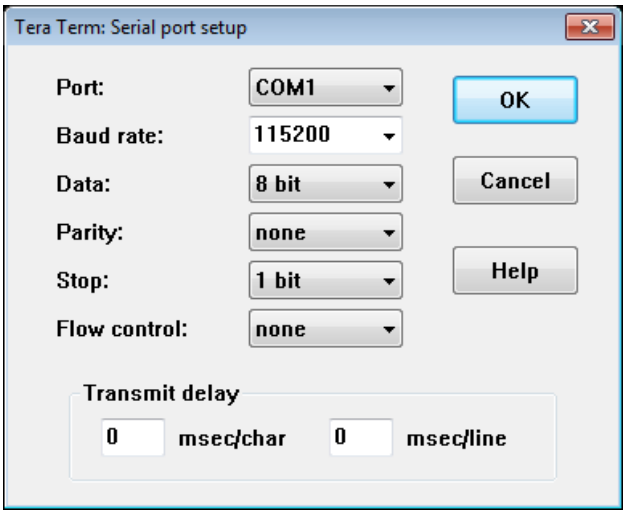


Figure 7: Terminal configuration using the example of "TeraTerm"

7.3 Ethernet configuration of the PLCcore-iMX35

The main Ethernet configuration of the PLCcore-iMX35 takes place within the bootloader "U-Boot" and is taken on for all software components (Linux, PLC firmware, HTTP server etc.). The Ethernet configuration is carried out via the serial interface COM0. **Therefore, the "U-Boot" command prompt must be activated as described in section 7.2**. Table 8 lists up "U-Boot" commands necessary for the Ethernet configuration of the PLCcore-iMX35.

Table 8: "U-Boot" configuration commands of the PLCcore-iMX35

Configuration	Command	Remark
MAC address	setenv ethaddr <xx:xx:xx:xx:xx:xx>	The MAC address worldwide is a clear identification of the module and is assigned by the producer. **It should not be modified by the user.**
IP address	setenv ipaddr <xxx.xxx.xxx.xxx>	This command sets the local IP address of the PLCcore-iMX35. The IP address is to be defined by the network administrator.
Network mask	setenv netmask <xxx.xxx.xxx.xxx>	This command sets the network mask of the PLCcore-iMX35. The network mask is to be defined by the network administrator.
Gateway address	setenv gatewayip <xxx.xxx.xxx.xxx>	This command defines the IP address of the gateway which is to be used by the PLCcore-iMX35. The gateway address is set by the network administrator. **Advice:** If PLCcore-iMX35 and Programming PC are located within the same sub-net, defining the gateway address may be skipped and value "0.0.0.0" may be used instead.
Saving the configuration	saveenv	This command saves active configurations in the flash of the PLCcore-iMX35.

Modified configurations may be verified again by entering "*printenv*" in the "U-Boot" command prompt. Active configurations are permanently saved in the Flash of the PLCcore-iMX35 by command

saveenv

Modifications are adopted upon next Reset of the PLCcore-iMX35.

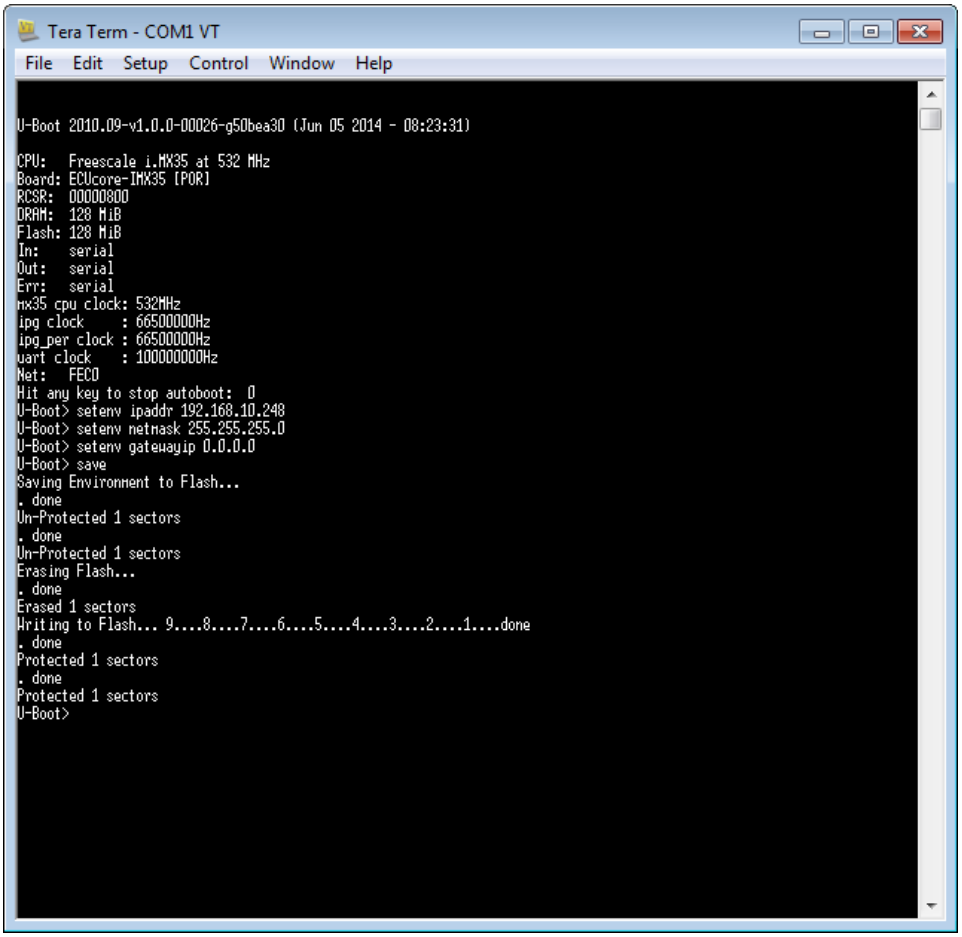


Figure 8: Ethernet configuration of the PLCcore-iMX35

After the configuration is finished and according to section 7.2, all conditions for a Linux Autostart must be re-established.

Upon Reset (e.g. pushbutton S601 on the Development Board) the module starts using the active configurations.

Advice: After the configuration is finished, the serial connection between PC and PLCcore-iMX35 is no longer necessary.

7.4 Boot configuration of the PLCcore-iMX35

The PLCcore-iMX35 is configured so that after Reset the PLC firmware starts automatically. Therefore, all necessary commands are provided by the start script ***"/home/etc/autostart"***. Hence, the required environment variables are set and drivers are booted.

If required, the start script ***"/home/etc/autostart"*** may be complemented by further entries. For example, by entering command *"pureftp"*, the FTP server is called automatically when the PLCcore-iMX35 is booted. The script can be edited directly on the PLCcore-iMX35 in the FTP client *"WinSCP"* (compare section 7.1) using pushbutton *"F4"* or *"F4 Edit"*.

7.5 Predefined user accounts

All user accounts listed in Table 9 are predefined upon delivery of the PLCcore-iMX35. Those allow for a login to the command shell (serial RS232 connection or Telnet) and at the FTP server of the PLCcore-iMX35.

Table 9: Predefined user accounts of the PLCcore-iMX35

User name	Password	Remark
PlcAdmin	Plc123	Predefined user account for the administration of the PLCcore- iMX35 (configuration, user administration, software updates etc.)
root	Sys123	Main user account ("root") of the PLCcore-iMX35

7.6 Login to the PLCcore-iMX35

7.6.1 Login to the command shell

In some cases the administration of the PLCcore-iMX35 requires the entry of Linux commands in the command shell. Therefore, the user must be directly logged in at the module. There are two different possibilities:

- Logging in is possible with the help of a **Terminal program** (e.g. HyperTerminal or TeraTerm, see section 7.1) via the serial interface **COM0** of the PLCcore-iMX35 – analog to the procedure described for the Ethernet configuration in section 7.2. **For the configuration of the terminal settings pay attention to only use "CR" (carriage return) as end-of-line character.** Login with user name and password is not possible for "CR+LF" (carriage return + line feed)!

- Alternatively, the login is possible using a **Telnet client** (e.g. Telnet or also TeraTerm) via the Ethernet interface **ETH0** of the PLCcore-iMX35.

For logging in to the PLCcore-iMX35 via the Windows standard Telnet client, the command "*telnet*" must be called by using the IP address provided in section 7.2, e.g.

```
telnet  192.168.10.248
```

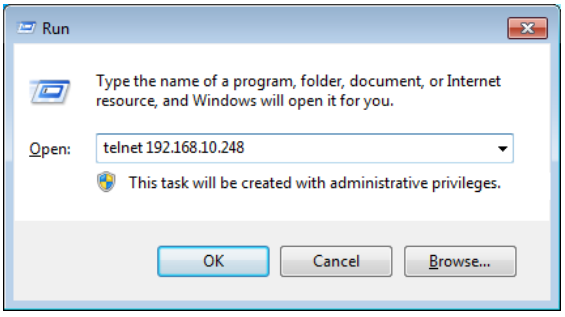


Figure 9: Calling the Telnet client in Windows

Logging in to the PLCcore-iMX35 is possible in the Terminal window (if connected via COM0) or in the Telnet window (if connected via ETH0). The following user account is preconfigured for the administration of the module upon delivery of the PLCcore-iMX35 (also compare section 7.5):

User: *PlcAdmin*
Password: *Plc123*

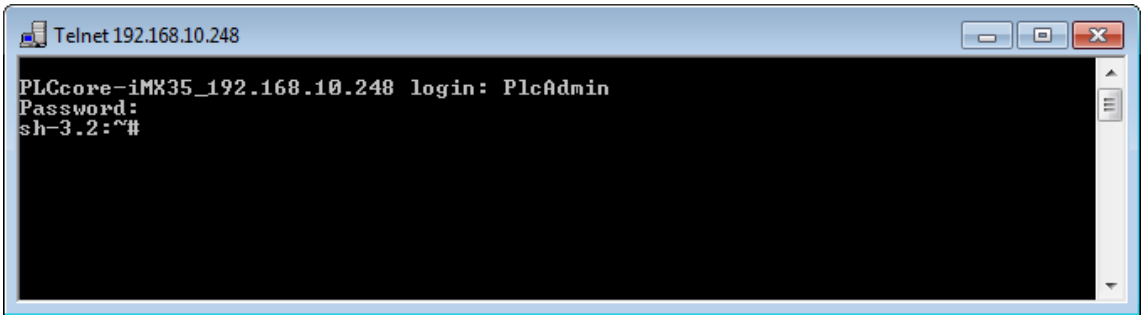


Figure 10: Login to the PLCcore-iMX35

Figure 10 exemplifies the login to the PLCcore-iMX35 using a Windows standard Telnet client.

7.6.2 Login to the FTP server

The PLCcore-iMX35 has available a FTP server (FTP Daemon) that allows file exchange with any computer (up- and download of files). Due to security and performance reasons, the FTP server is deactivated by default and must be started manually if required. Therefore, the user must first be logged in to the command shell of the PLCcore-iMX35 following the procedures described in section 7.6.1. Afterwards, the following command must be entered in the Telnet or Terminal window:

```
pureftp
```

Figure 11 illustrates an example for starting the FTP server.

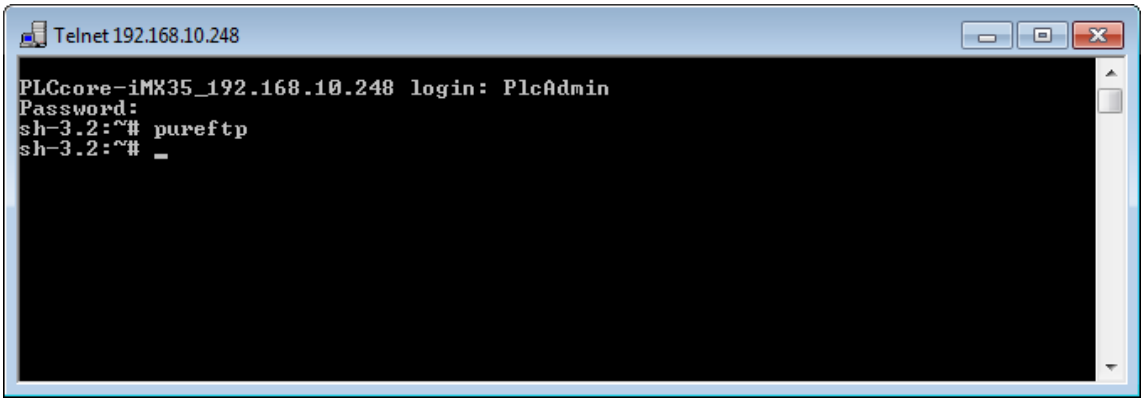


Figure 11: Starting the FTP server

Advice: By entering command *"pureftp"* in the start script ***"/home/etc/autostart"***, the FTP server may be called automatically upon boot of the PLCcore-iMX35 (see section 7.4).

"WinSCP" - which is available as open source - would be suitable as FTP client for the computer (see section 7.1). It consists of only one EXE file, needs no installation and may be started immediately.

After program start, dialog *"WinSCP Login"* appears (see Figure 12) and must be adjusted according to the following configurations:

File protocol: FTP
Host name: IP address for the PLCcore-iMX35 as set in section 7.3
User name: *PlcAdmin* (for predefined user account, see section 7.5)
Password: *Plc123* (for predefined user account, see section 7.5)

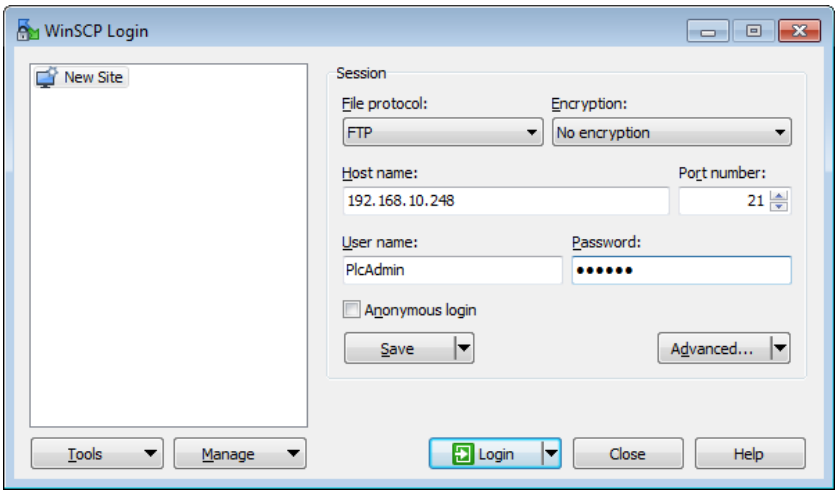


Figure 12: Login settings for WinSCP

After using pushbutton *"Login"*, the FTP client logs in to the PLCcore-iMX35 and lists up the active content of directory *"/home"* in the right window. Figure 13 shows FTP client *"WinSCP"* after successful login to the PLCcore-iMX35.

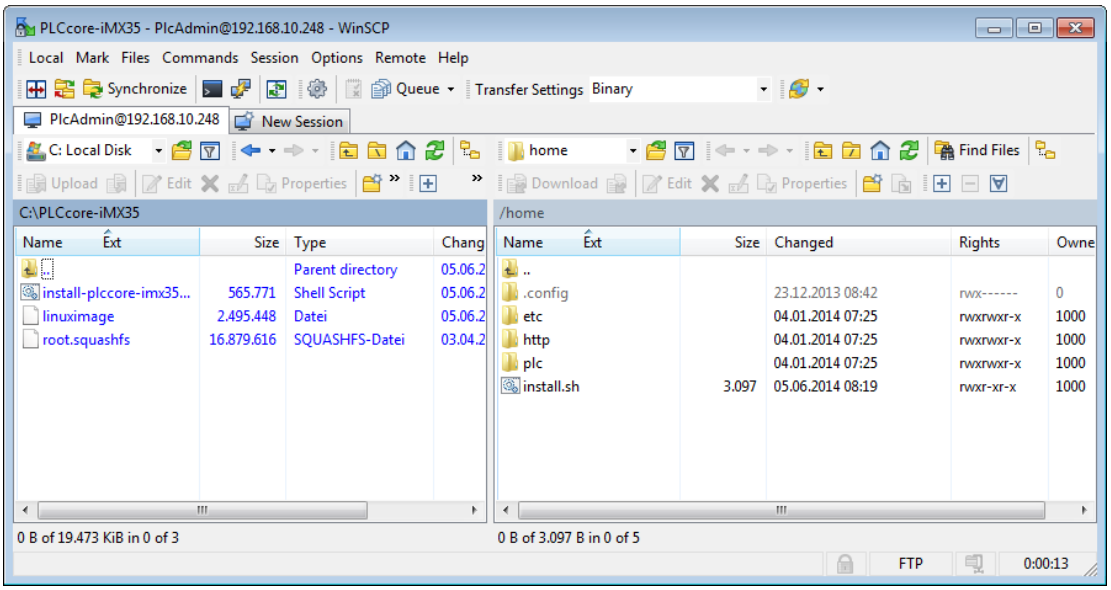


Figure 13: FTP client for Windows "WinSCP"

After successful login, configuration files on the PLCcore-iMX35 may be edited by using pushbuttons *"F4"* or *"F4 Edit"* within the FTP client *"WinSCP"* (select transfer mode *"Text"*). With the help of pushbutton *"F5"* or *"F5 Copy"*, files may be transferred between the computer and the PLCcore-

iMX35, e.g. for data backups of the PLCcore-iMX35 or to transfer installation files for firmware updates (select transfer mode *"Binary"*).

7.7 Adding and deleting user accounts

Adding and deleting user accounts requires the login to the PLCcore-iMX35 as described in section 7.6.1.

Adding a new user account takes place via Linux command *"adduser"*. In embedded systems such as the PLCcore-iMX35, it does not make sense to open a directory for every user. Hence, parameter *"-H"* disables the opening of new directories. By using parameter *"-h /home"* instead, the given directory *"/home"* is rather assigned to the new user. To open a new user account on the PLCcore-iMX35, Linux command *"adduser"* is to be used as follows:

```
adduser  -h /home  -H  -G <group>  <username>
```

Figure 14 exemplifies adding a new account on the PLCcore-iMX35 for user *"admin2"*.

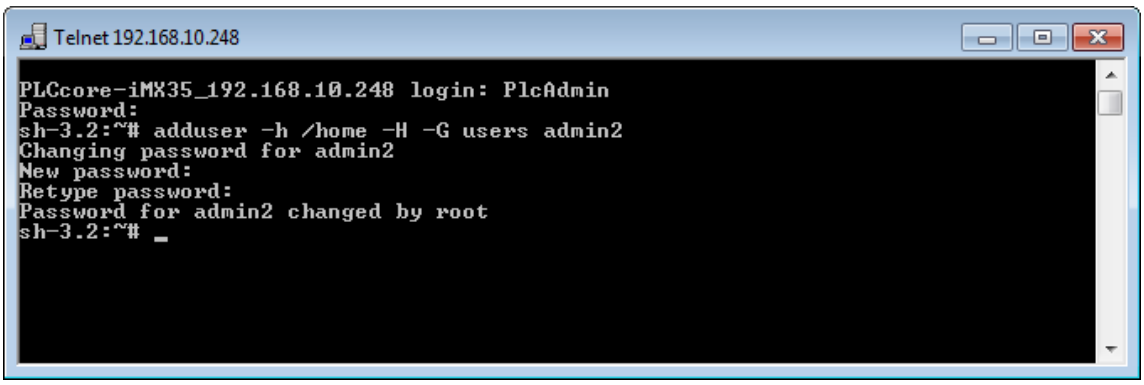


Figure 14: Adding a new user account

To **delete** an existing user account from the PLCcore-iMX35, Linux command *"deluser"* plus the respective user name must be used:

```
deluser  <username>
```

7.8 How to change the password for user accounts

Changing the password for user accounts requires login to the PLCcore-iMX35 as described in section 7.6.1.

To change the password for an existing user account on the PLCcore-iMX35, Linux command *"passwd"* plus the respective user name must be entered:

```
passwd <username>
```

Figure 15 exemplifies the password change for user *"PlcAdmin"*.

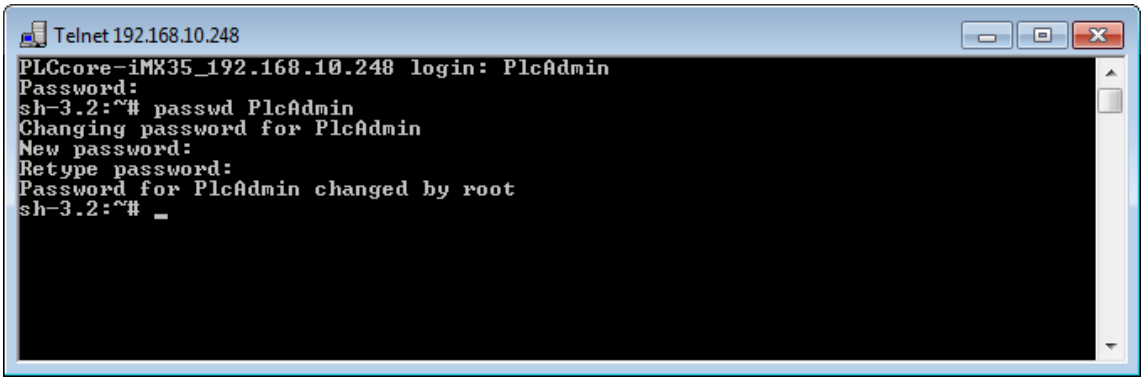


Figure 15: Changing the password for a user account

7.9 Setting the system time

Setting the system time requires login to the PLCcore-iMX35 as described in section 7.6.1.

There are two steps for setting the system time of the PLCcore-iMX35. At first, the current date and time must be set using Linux command *"date"*. Afterwards, by using Linux command *"hwclock -w"* the system time is taken over into RTC module of the PLCcore-iMX35.

Linux command *"date"* is structured as follows:

```
date [options] [YYYY.]MM.DD-hh:mm[:ss]
```

Example:

```
date   2014.06.05-14:00:35
          |  |  |  |  |  |
          |  |  |  |  |  +--- Second
          |  |  |  |  +------ Minute
          |  |  |  +--------- Hour
          |  |  +------------ Day
          |  +--------------- Month
          +------------------ Year
```

To set the system time of the PLCcore-iMX35 to 2014/06/05 and 14:00:35 (as shown in the example above), the following commands are necessary:

```
date  2014.06.05-14:00:35
hwclock  -w
```

The current system time is displayed by entering Linux command *"date"* (without parameter). The Linux command *"hwclock -r"* can be used to recall current values from the RTC. By using *"hwclock -s"*, the current values of the RTC are taken over as system time for Linux (synchronizing the kernel with the RTC). Figure 16 exemplifies setting and displaying the system time.

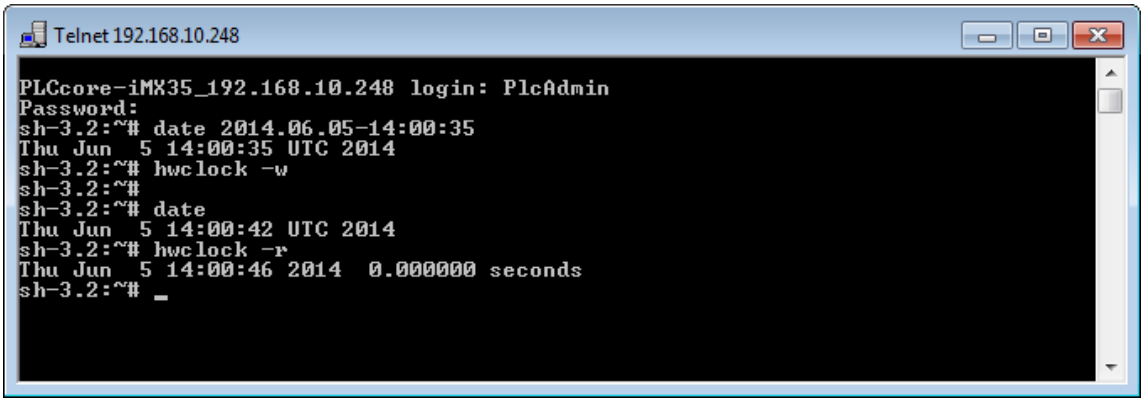


Figure 16: Setting and displaying the system time

Upon start of the PLCcore-iMX35, date and time are taken over from the RTC and set as current system time of the module. Therefore, Linux command *"hwclock -s"* is necessary which is included in start script ***"/etc/init.d/hwclock"***.

7.10 File system of the PLCcore-iMX35

Pre-installed Embedded Linux on the PLCcore-iMX35 provides part of the system memory in form of a file system. Being usual for embedded systems, most of this file system is “read/only” which means that changes to this part can only be made by creating a new Linux-Image for the PLCcore-iMX35. The advantage hereby is the resistance of a read/only file system against damages in case of power breakdowns. Those occur relatively often in embedded systems because embedded systems are usually simply turned off without previous shutdown.

Table 10 lists up writable paths of the file system during runtime. Path **"/home"** comprises a flash disk that provides part of the on-board flash memory of the PLCcore-iMX35 as file system. This path is used to store all files modifiable and updatable by the user, e.g. configuration files, PLC firmware and PLC program files that have been loaded onto the module. Directory **"/tmp"** is appropriately sized to function as temporary buffer for FTP downloads of firmware archives for PLC software updates (see section 7.12.1).

Table 10: File system configuration of the PLCcore-iMX35

Path	Size	Description
/home	95516 kByte	Flash disk to permanently store files modifiable and updatable by the user (e.g. configuration files, PLC firmware, PLC program, files for Target Visualization), data preservation in case of power breakdown
/tmp	8192 kByte	RAM disk, suitable as intermediate buffer for FTP downloads, but no data preservation in case of power breakdown
/var	62640 kByte	RAM disk which is used by the system to store temporary files, no data preservation in case of power breakdown
/mnt		Target for integrating remote directories, it is not part of the PLCcore-iMX35 standard functionality

Sizes of file system paths that are configured or still available can be identified by using the Linux command "df" ("DiskFree") – see Figure 17.

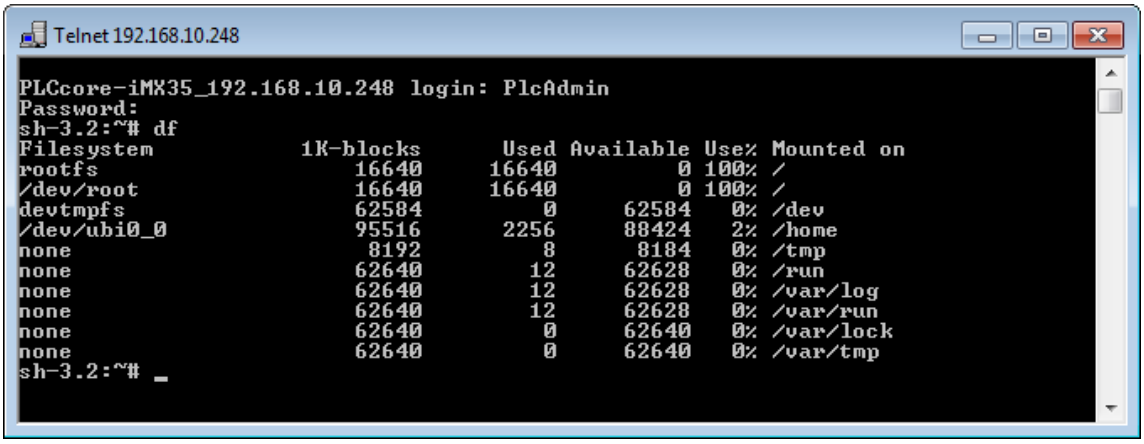


Figure 17: Display of information about the file system

Particular information about the system login and handling the Linux command shell of the PLCcore-iMX35 is given attention in section 7.6.

7.11 Calibration of the Touchscreen

The PLCcore-iMX35 has no on-board touch controller. Hence, an external touch controller is necessary to use resistive Touchscreens. Touchscreen and touch controller have to be adjusted – that means calibrated – to another before its first use. Without a calibration, the Touchscreen works extremely imprecise which normally makes a correct operation impossible.

7.11.1 Automatic Test of Touchscreen Calibration

An extensive calibration is needed before using the Touchscreen. During booting the PLC system, the device software can check, whether the required calibration of the Touchscreen has already been undertaken. Therefore is tested, if the file *"/home/etc/pointercal"* exists and if this file has a size greater 0 byte. If this condition is not fulfilled, the appropriate calibration program *"ts_calibrate"* is executed before starting the PLC firmware.

As the PLCcore-iMX35 supports displays with and without Touchscreen, an automatic check of the Touchscreen calibration can be enabled or disabled as desired within the configuration settings of the module. The particular calibration occurs by means of the environment variable *"check_tscalibfile"* of the bootloader “U-Boot”. To set this variable, the command prompt relating to the “U-Boot” has to be enabled first, as described in section 7.2. Table 11 lists all commands for enabling / disabling the automatic control.

Table 11: Configuration for automatically checking of Touchscreen calibration

Command	Setting
setenv check_tscalibfile on saveenv	automatically checking of Touchscreen calibration activated, in case that file *"/home/etc/pointercal"* doesn't exist (or has a size of 0 byte), the calibration program *"ts_calibrate"* will be launched automatically
setenv check_tscalibfile off saveenv	automatically checking of Touchscreen calibration deactivated, existing of file *"/home/etc/pointercal"* will not be checked

Advice: The command *"saveenv"*, also stated in Table 11, is necessary to save the modified configuration persistently in the Flash of the PLCcore-iMX35.

7.11.2 Manually calibration of the Touchscreen

The manually calibration of the Touchscreen occurs interactively, by the operators click on the markings (“Reticles”) given on the display. The calibration program needed for it is started from the command line, which requires login to the PLCcore-iMX35 as described in section 7.6.1. After that, the following command has to be entered in the Telnet- or Terminal-window:

```
ts_calibrate
```

In the course of the calibration sequence, 5 markings (“Reticles”, in each corner and in the middle) are shown one after another on the display, which are to click by the user. The more exact the shown markings are clicked, the higher the achievable accuracy during the later operation of the Touchscreen. It is therefore recommended to use a touchpen or stylus during calibration as it is used for Handhelds, PDAs or drawing tablets.

After finishing calibration, the calibration data are stored in file *"/home/etc/pointercal".* In case this file gets lost, e.g. through reformatting of the flash-disk, the calibration has to be carried out again.

Advice: The Development Kit PLCcore-iMX35 is delivered completely calibrated. A recalibration is only necessary in exceptional cases (e.g. after a change of display with integrated Touchscreen).

7.12 Software update of the PLCcore-iMX35

All necessary firmware components to run the PLCcore-iMX35 are already installed on the module upon delivery. Hence, firmware updates should only be required in exceptional cases, e.g. to input new software that includes new functionality.

7.12.1 Updating the PLC firmware

PLC firmware indicates the run time environment of the PLC. ***PLC firmware*** can only be generated and modified by the producer; **it is not identical with the PLC user program** which is created by the PLC user. The PLC user program is directly transferred from the *CODESYS* programming environment onto the module. No additional software is needed.

Updating the PLC firmware requires login to the command shell of the PLCcore-iMX35 as described in section 7.6.1 and login to the FTP server as described in section 7.6.2.

Updating the PLC firmware takes place via a self-extracting firmware archive that is transferred onto the PLCcore-iMX35 via FTP. After starting the FTP server on the PLCcore-iMX35 (command *"pureftp"*, see section 7.6.2), the respective firmware archive can be transferred into directory *"/tmp"* of the PLCcore-iMX35 (see Figure 18).

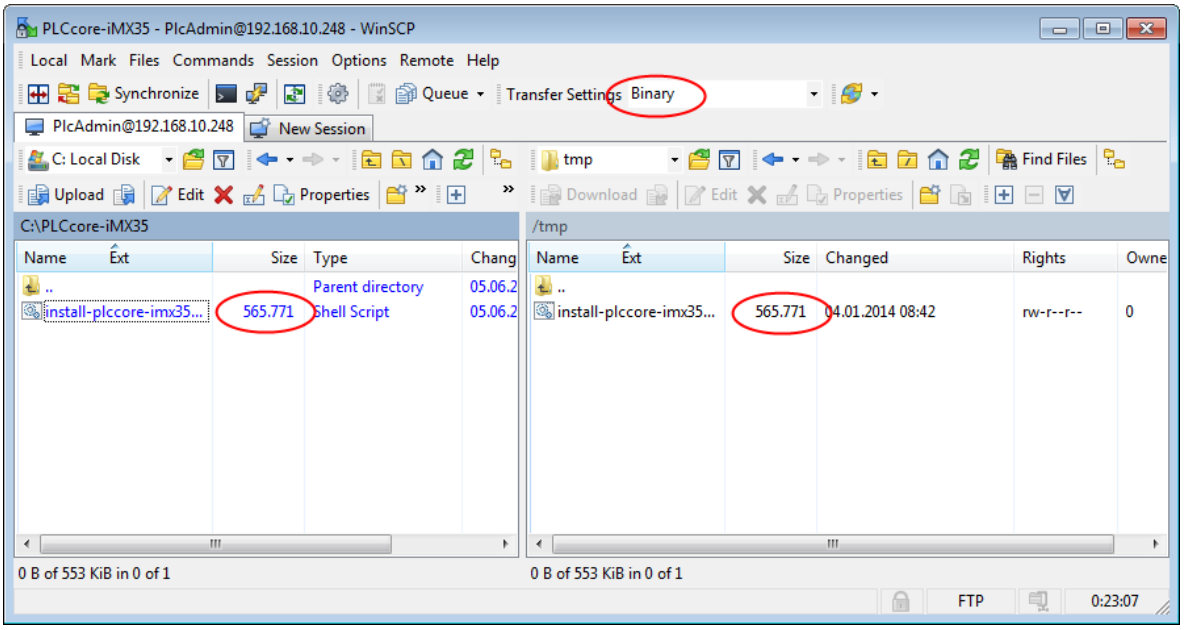


Figure 18: File transfer in FTP client "WinSCP"

Important: To transfer the firmware archive via FTP, transfer type *"Binary"* must be chosen. If FTP client *"WinSCP"* is used, the appropriate transfer mode is to be chosen from the menu bar. After downloading the firmware archive, it must be checked if the file transferred to the PLCcore-iMX35 has the exact same size as the original file on the computer (compare Figure 18). Any differences in that would indicate a mistaken transfer mode (e.g. *"Text"*). In that case the transfer must be repeated using transfer type *"Binary"*.

After downloading the self-extracting archive, the PLC firmware must be installed on the PLCcore-iMX35. Therefore, the following commands are to be entered in the Telnet window:

```
cd  /tmp
chmod  +x  install-plccore-imx35-CODESYS.sh
./ install-plccore-imx35-CODESYS.sh
```

Advice: The command shell of the PLCcore-iMX35 is able to automatically complete names if the Tab key is used ("tab completion"). Hence, it should be sufficient to enter the first letters of each file name and the system will complement it automatically. For example, *"./ins"* is completed to *"./install-plccore-imx35-CODESYS.sh"* if the Tab key is used.

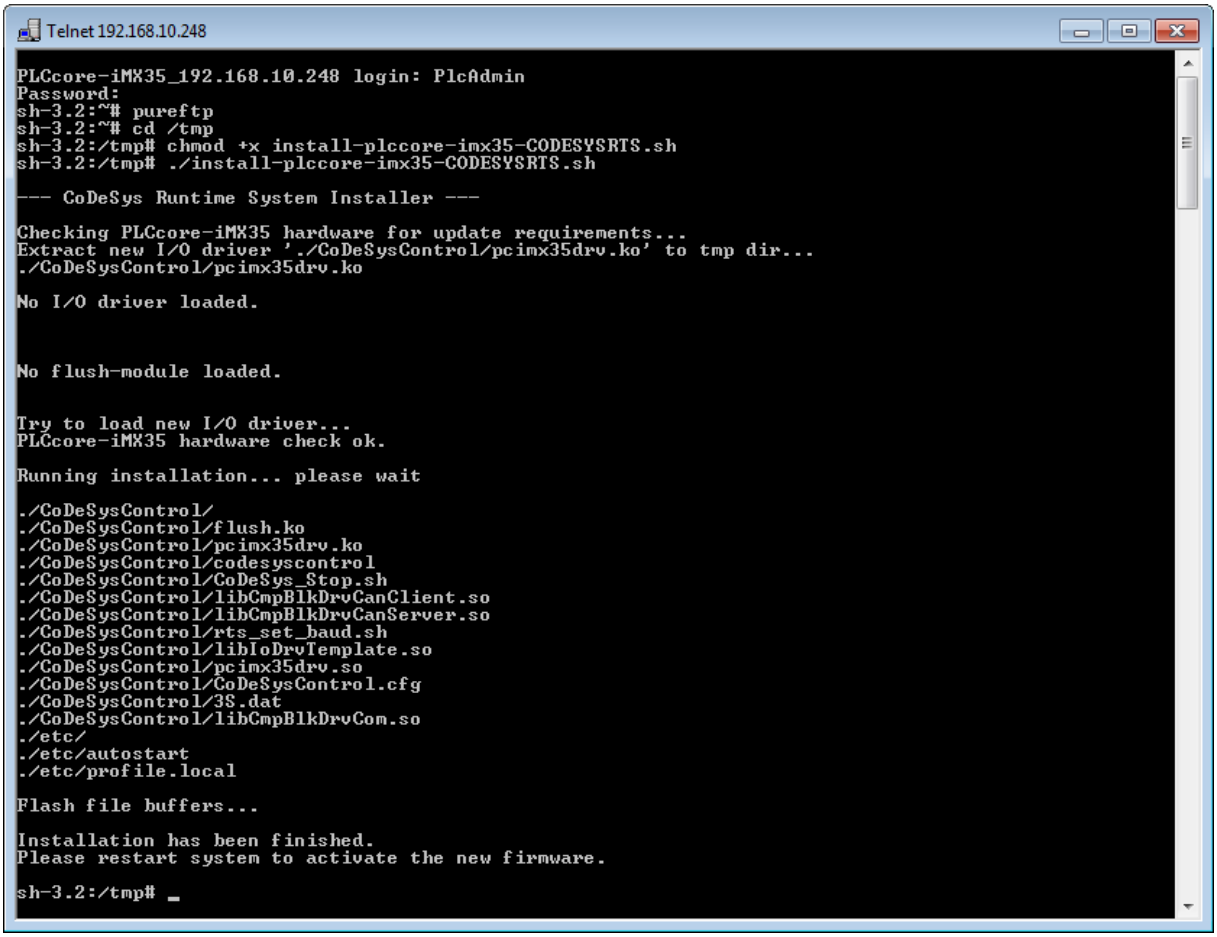


Figure 19: Installing PLC firmware on the PLCcore-iMX35

Figure 19 exemplifies the installation of PLC firmware on the PLCcore-iMX35. After Reset the module is started using the updated firmware.

7.12.2 How to update the Linux-Image

Updating the Linux-Image takes place via TFTP (**T**rivial **FTP**) within Linux bootloader *"U-Boot"*. Therefore, an appropriate TFTP server is necessary on the computer, e.g. freeware *"TFTPD32"* (compare section 7.1). The program consists of only one EXE file that requires no installation and can be run immediately. After the program start, an appropriate working directory ("Current Directory") should be created by clicking on pushbutton *"Browse"* (e.g. *"C:\PLCcore-iMX35"*). The image files for the PLCcore-iMX35 must be located in this directory (*"linuximage"* and *"root.squashfs"*).

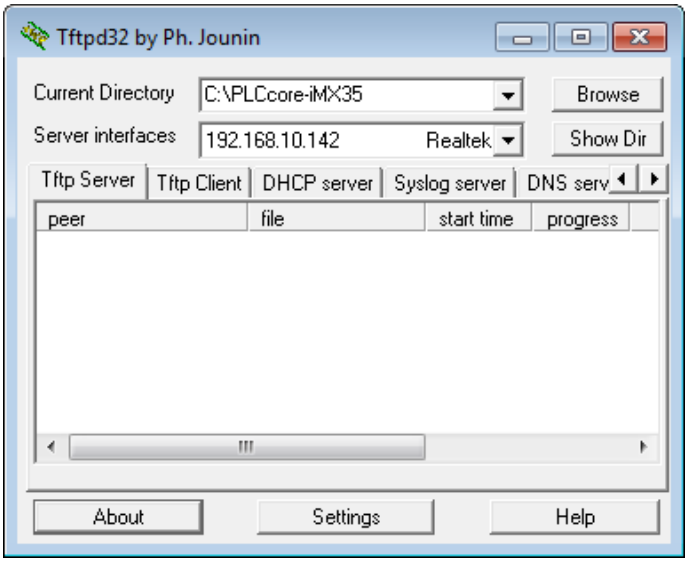


Figure 20: TFTP server for Windows "TFTPD32"

A TFTP download of the image files **requires** that the **Ethernet configuration** of the PLCcore-iMX35 is **completed** according to procedures describes in **section 7.3**. To update the Linux-Image it is necessary to have available another serial connection to the PLCcore-iMX35 in addition to the Ethernet connection. All configurations for the terminal program as described in section 7.2 apply (115200 Baud, 8 Data bit, 1 Stop bit, no parity and no flow control).

Updating the Linux-Image of the PLCcore-iMX35 is only possible if Linux is not running. Hence, Linux Autostart must be disabled prior to the updating process and "U-Boot" command prompt must be used instead. Procedures are described in section 7.2.

After Reset (e.g. pushbutton S601 on the Development Board), the "U-Boot" command prompt answers. To update the Linux-Image the following commands must be entered according to the following sequence:

Table 12: Command sequence to update the Linux-Image on the PLCcore-iMX35

Command	Meaning
setenv serverip <host_ip_addr>	Setting the IP address of the TFTP server. If *"TFTPD32"* is used, the address is shown in field "Server Interface" on the PC.
tftp linuximage	Downloading the Linux-Image from the Development PC onto the PLCcore-iMX35
erase nor0,4	Erase the Flash area, needed by Linux-Image
cp.b ${fileaddr} 0xa00e0000 ${filesize}	Saving the Linux-Image in the Flash of the PLCcore-iMX35
tftp root.squashfs	Downloading the Root File System from the Development PC onto the PLCcore-iMX35
erase nor0,5	Erase the Flash area, needed by Root File System
cp.b ${fileaddr} 0xa04e0000 ${filesize}	Saving the Root File System in the Flash of the PLCcore-iMX35

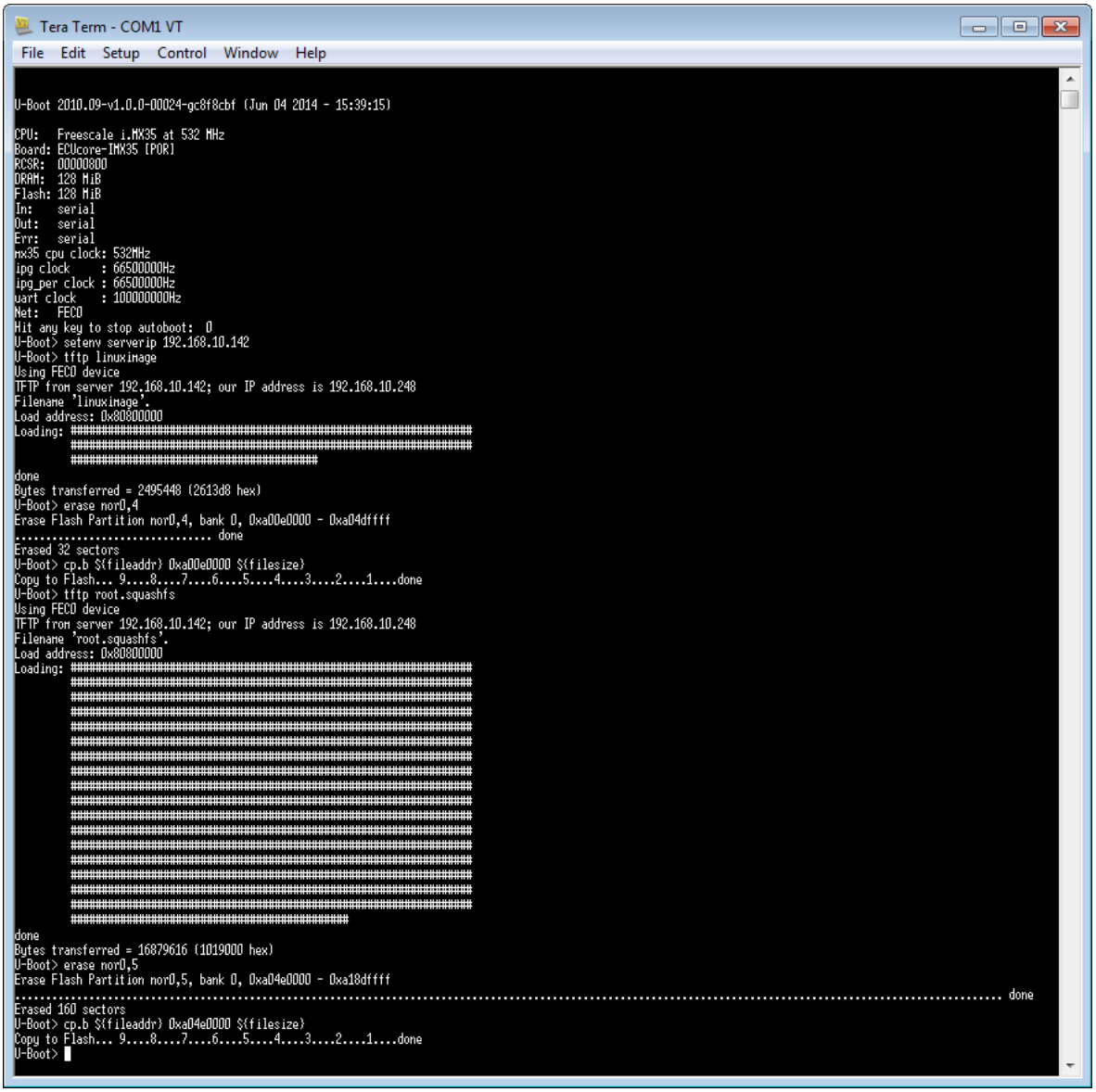


Figure 21: Downloading the Linux-Image to the PLCcore-iMX35

After completing the configuration, conditions for a Linux Autostart must be reestablished according to instructions in section 7.2.

After Reset is activated (e.g. pushbutton S601 on the Development Board), the PLCcore-iMX35 starts automatically using the current Linux-Image.

Advice: After the configuration is finished, the serial connection between the computer and the PLCcore-iMX35 is no longer necessary.

Appendix A: Reference design for the PLCcore-iMX35

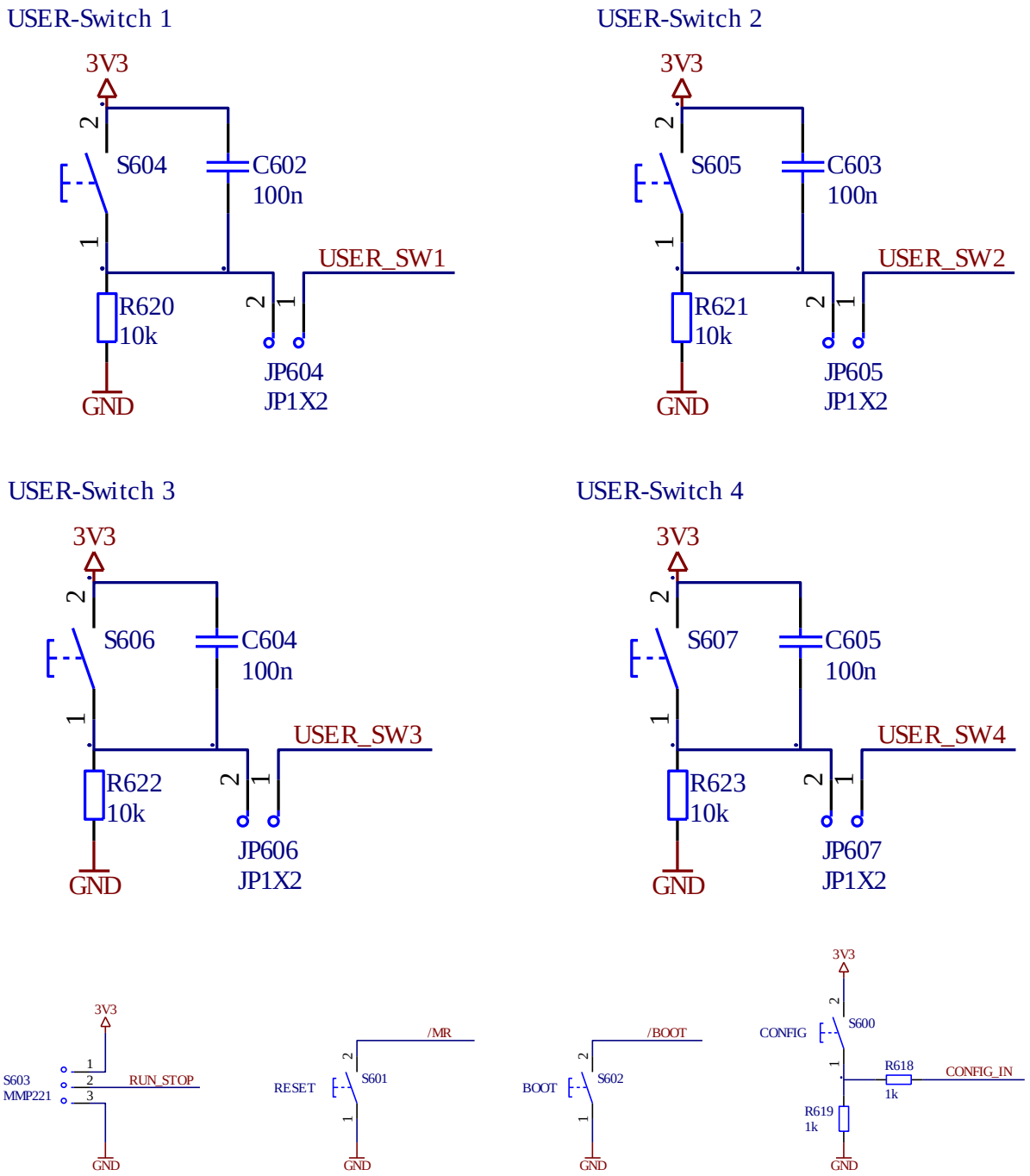


Figure 22: Reference design for User Controls

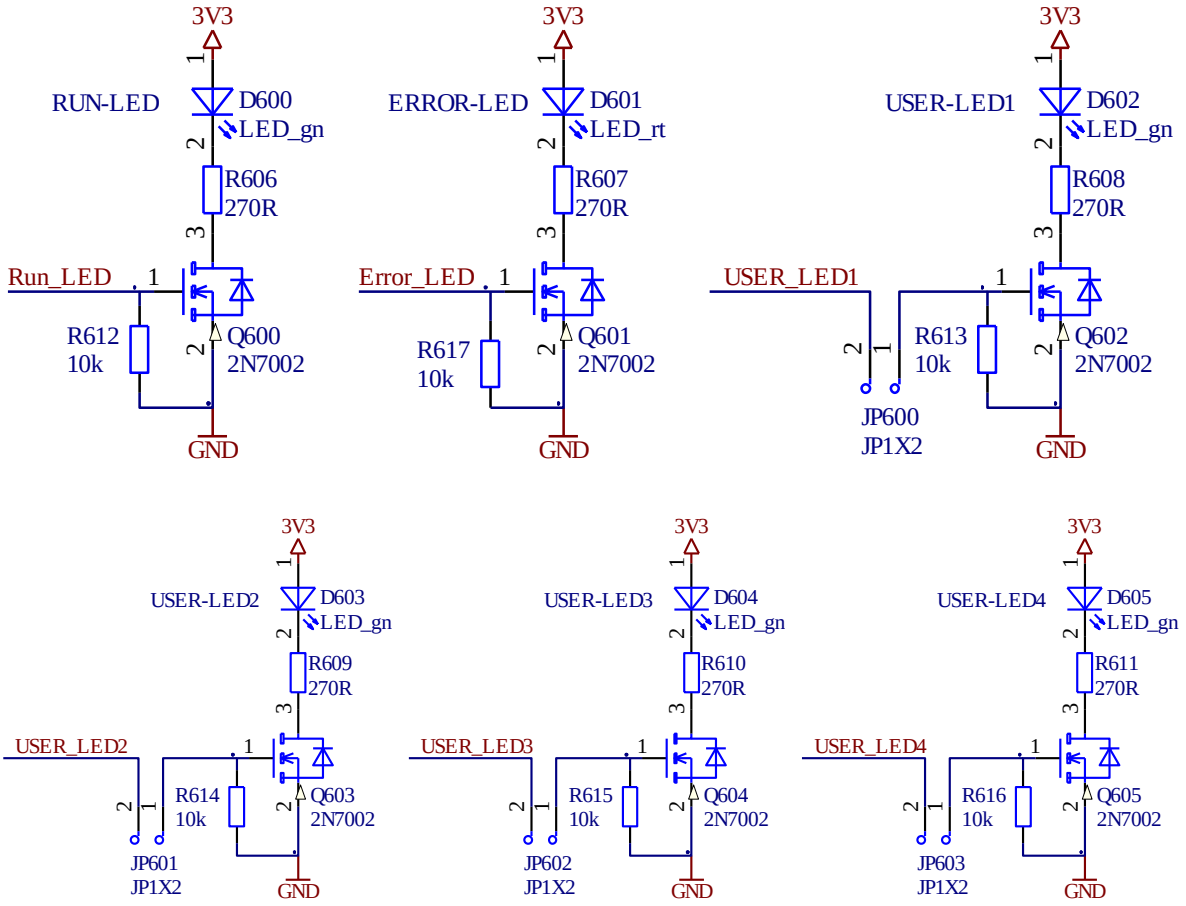


Figure 23: Reference design for LEDs

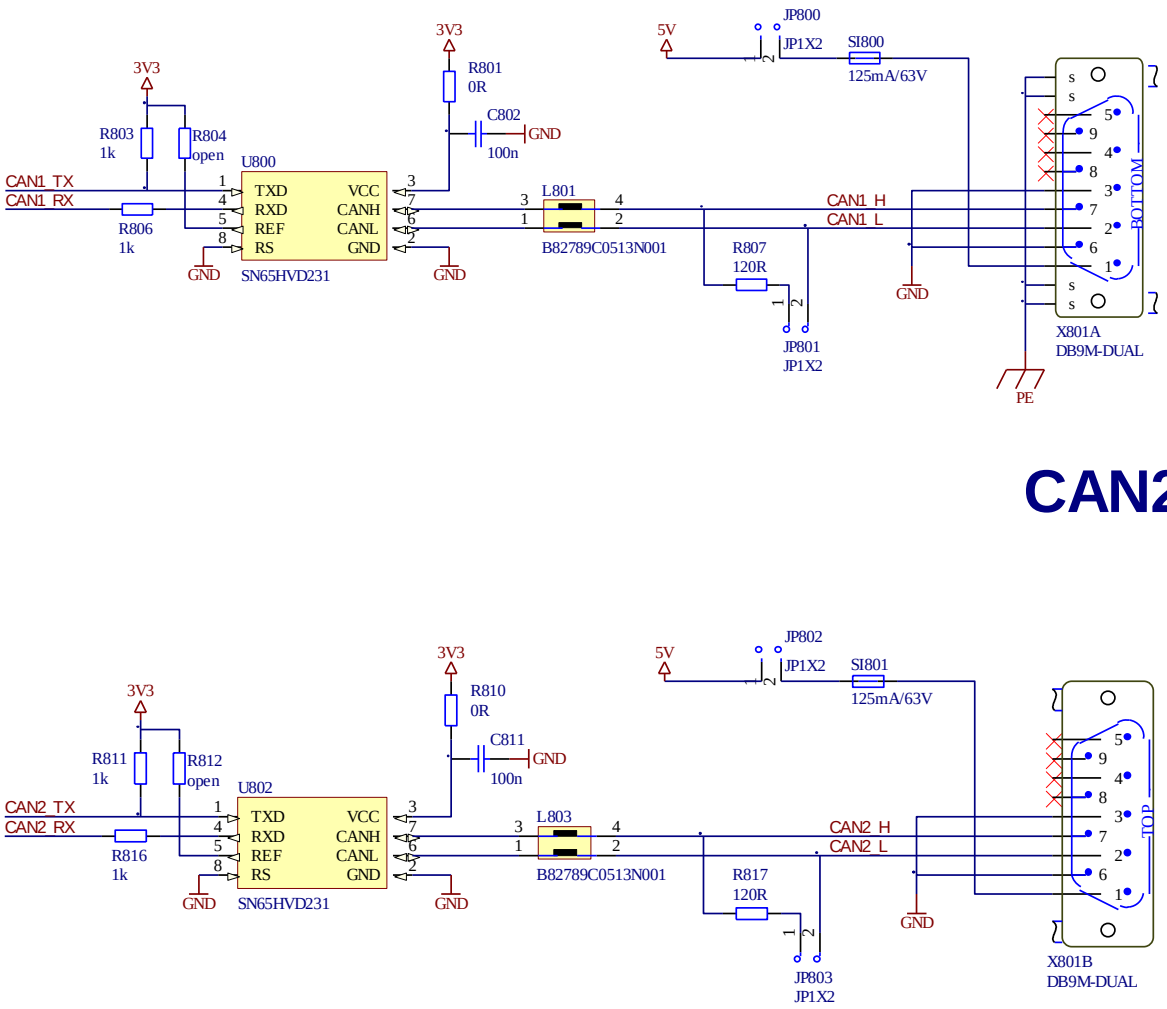


Figure 24: Reference design for interface circuits CAN

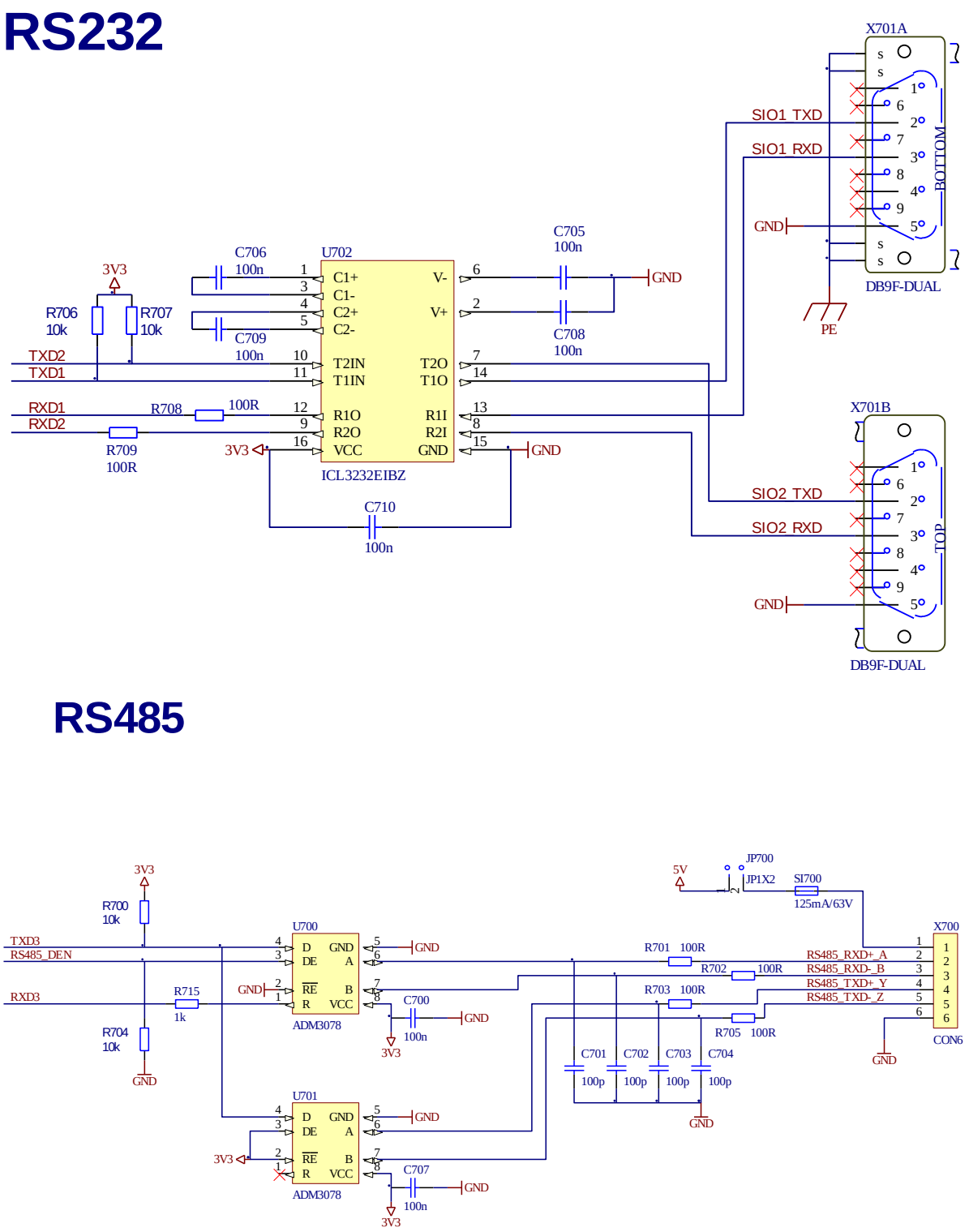


Figure 25: Reference design for interface circuits RS232/485

LAN

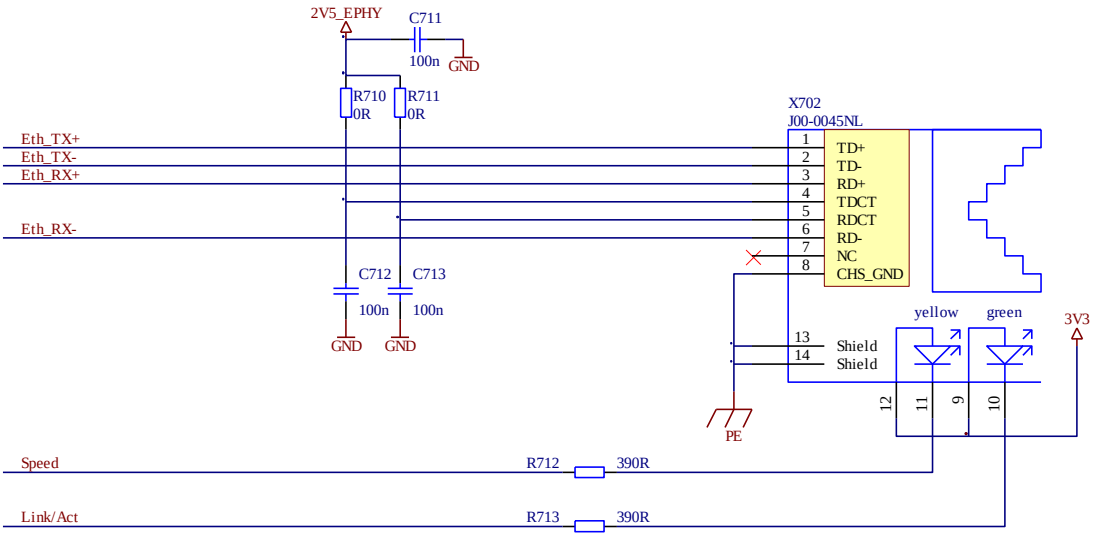
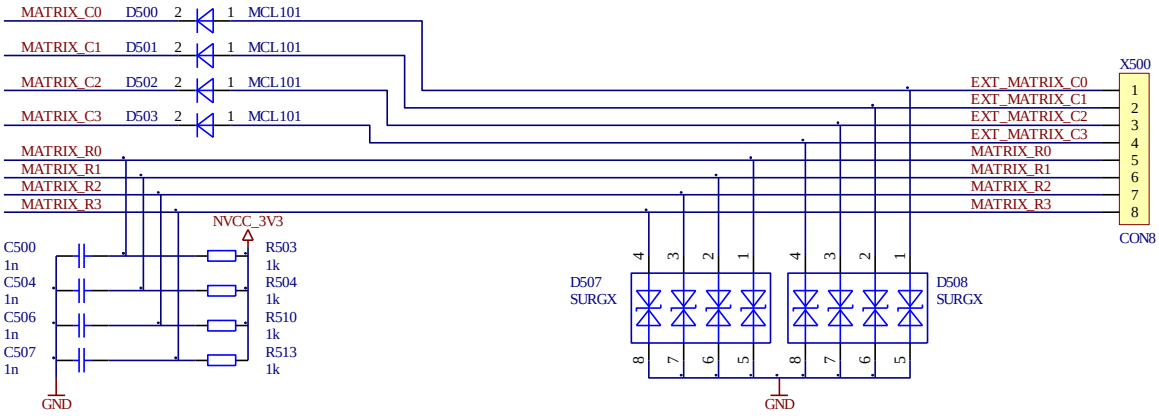


Figure 26: Reference design for interface circuit Ethernet

Keypad



Scroll wheel

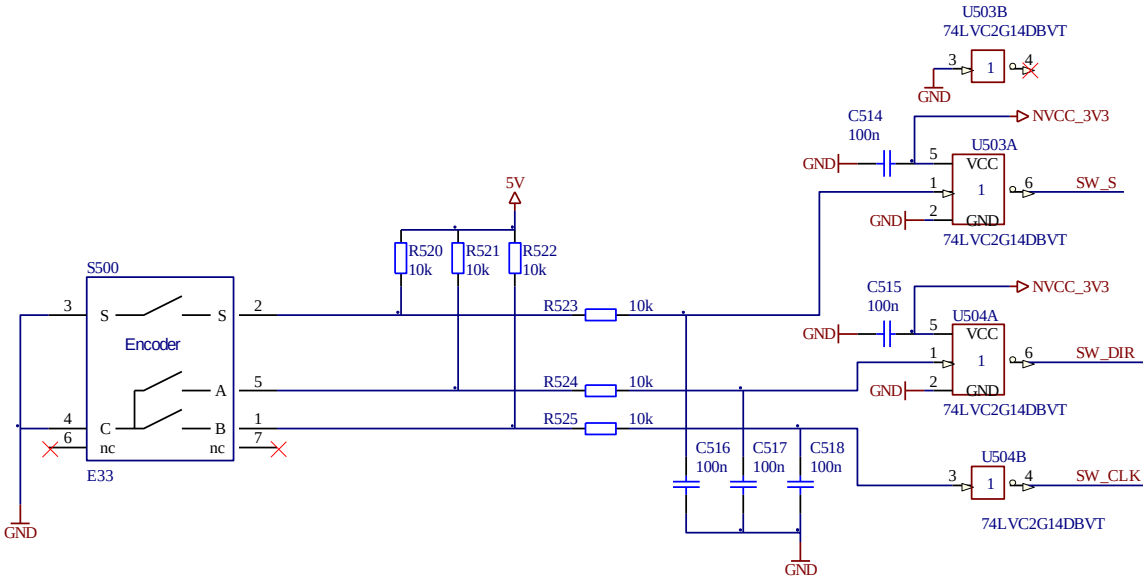


Figure 27: Reference design for Matrix Keypad and Scrollwheel

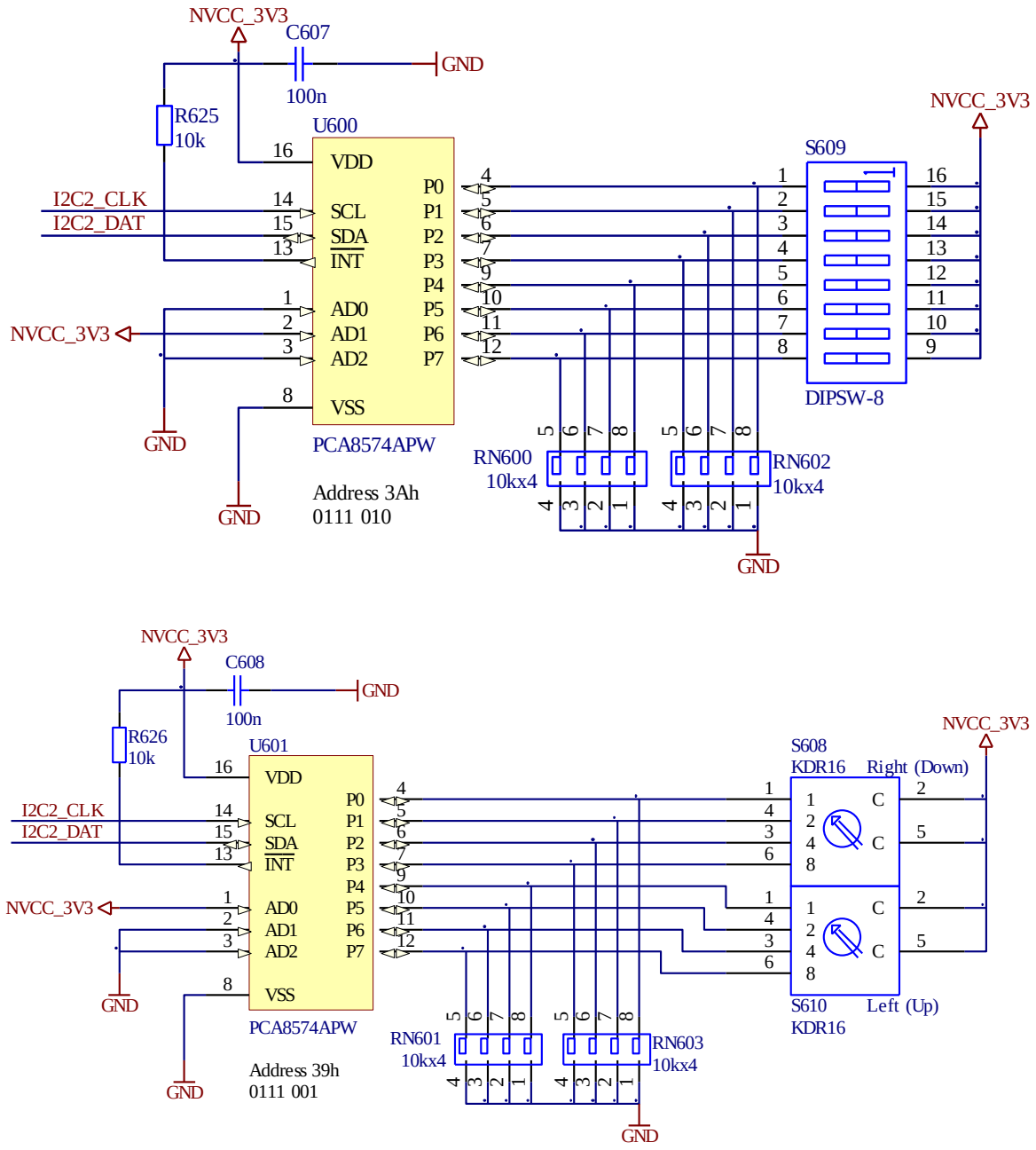


Figure 28: Reference design HEX- and DIP-Switches

Appendix B: GNU GENERAL PUBLIC LICENSE

The Embedded Linux used on the PLCcore-iMX35 is licensed under GNU General Public License, version 2. The entire license text is specified below.

The PLC system used and the PLC and C/C++ programs developed by the user are **not** subject to the GNU General Public License!

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software -- to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions;
type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items -- whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program `Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

Index

Document:	System Manual PLCcore-iMX35
Document number:	L1573e_1, May 2016

How would you improve this manual?

Did you detect any mistakes in this manual? page

Submitted by:

Customer number: _____

Name: _____

Company: _____

Address: _____

Please return your suggestions to:

SYS TEC electronic GmbH
Am Windrad 2
D - 08468 Heinsdorfergrund
GERMANY
Fax: +49 (0) 37 65 / 38600-4100
Email: info@systec-electronic.com