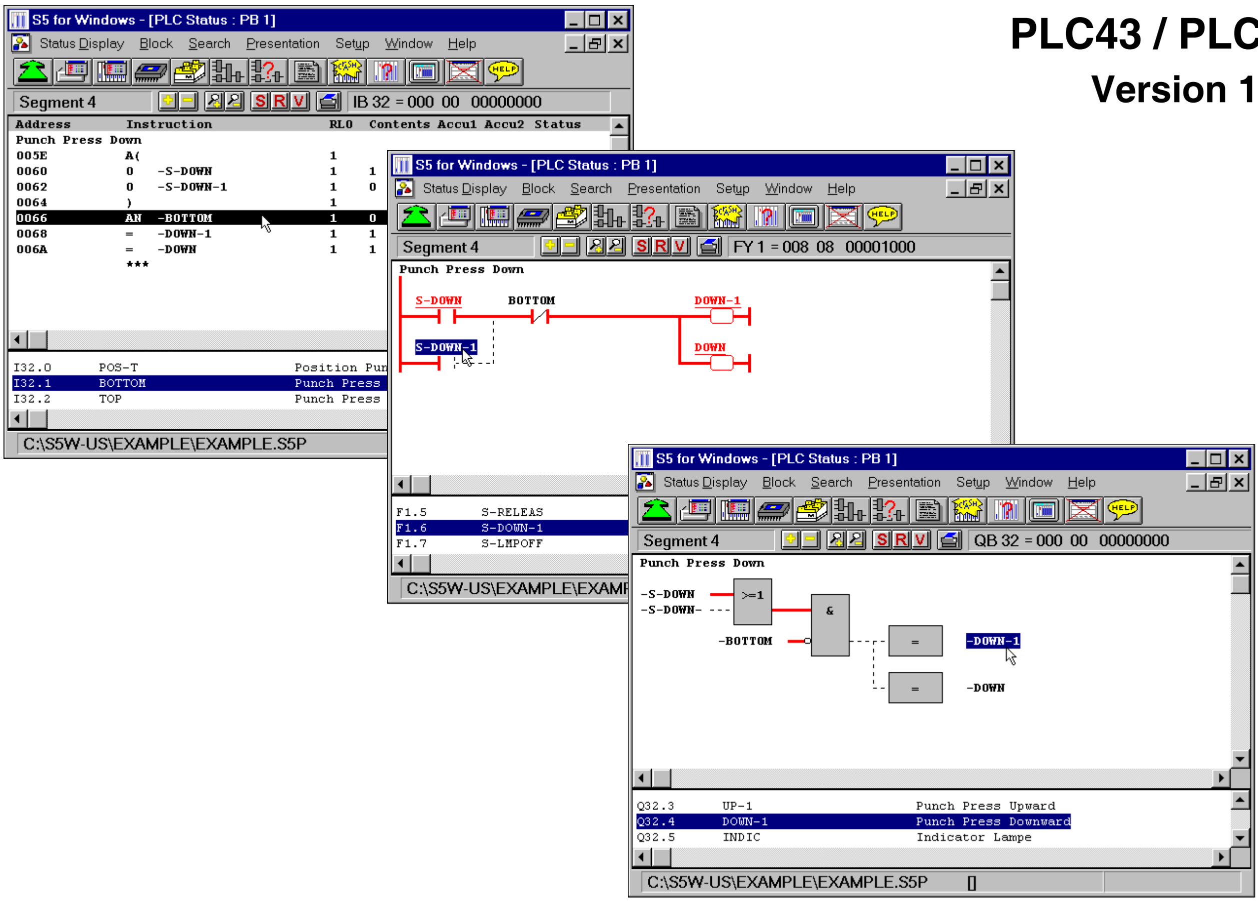
IBHsoftec

SoftPLC

PLC43 / PLC45

Version 1.6x



```
Address        Instruction                   RLO   Contents Accu1 Accu2  Status
Punch Press Down
005E           A(                             1
0060           O     -S-DOWN                  1     1
0062           O     -S-DOWN-1                1     0
0064           )                              1
0066           AN    -BOTTOM                  1     0
0068           =     -DOWN-1                  1     1
006A           =     -DOWN                    1     1
               ***

I32.0      POS-T          Position Pun
I32.1      BOTTOM         Punch Press
I32.2      TOP            Punch Press
```

```
Punch Press Down
S-DOWN        BOTTOM                      DOWN-1
S-DOWN-1                                  DOWN

F1.5       S-RELEAS
F1.6       S-DOWN-1
F1.7       S-LMPOFF
```

```
Punch Press Down
-S-DOWN     >=1
-S-DOWN-          &
-BOTTOM                 =     -DOWN-1
                        =     -DOWN

Q32.3      UP-1           Punch Press Upward
Q32.4      DOWN-1         Punch Press Downward
Q32.5      INDIC          Indicator Lampe
```

S5 for Windows

SoftPLC PLC43 / PLC45

User's Guide

Version 1.6x

Contents

1 *SoftPLC* Installation

The *SoftPLC*, **PLC 43 NT** and **PLC 45 NT**, require the Microsoft Windows NT Version 4.x / Windows 2000 operating system.

1.1 System Requirements

The performance of the *SoftPLC* is dependent on the PC hardware. In general, the following guidelines should be considered when selecting the PC hardware.

Large hard disks (>1 Gbyte, IDE, EIDE) very often have automatic temperature compensation. Hard disks equipped with such an adjustment will disable the system during adjustment. This will extend the execution time of the *SoftPLC* cycle. Tests have shown that SCSI hard disks without temperature compensation have the lowest impact on changing the execution time of the *SoftPLC* cycle. PCI SCSI controller configured as a bus master will reduce the CPU load to a minimum.

Network communication will also put a heavy load on the PC CPU. To reduce the load, PCI boards for are preferable to ISA boards.

Dual processor systems (a motherboard with two CPU's) are the best environment to execute the *SoftPLC* software. In such a system, the *SoftPLC* uses one of the processors almost exclusively. This results in a very constant *SoftPLC* cycle time. A dual processor system with two processors with a 100 MHz clock is much preferable to a single processor system with a 200MHz CPU clock.

To use the full power of the Windows NT 4.0 / Windows 2000 operating system, the PC should be equipped with at least 32 Mbyte of RAM.

The PC construction should be according to its intended use and operating environment.

The *SoftPLC* software (PLC 43 NT, PLC 45 NT) may be installed on any PC running Windows NT 4.x / Windows 2000.

To install the *SoftPLC*, a floppy disk drive with the disk format 1.44 Mbyte (HD) 3.5 inch is required.

To access the *SoftPLC* online with an external PC, an open serial port (COM1 - COM4) must be available.

1.2 *SoftPLC* Response Time

The Response Time of the *SoftPLC* is fixed.

Single processor systems

50% of the processor time is allocated to the *SoftPLC*. The *SoftPLC* is called every 2 ms.

If the execution time of the PLC program is less than one (1) ms, the PLC execution time is reduced accordingly (the *SoftPLC* takes less than 50% of the processor time).

Multi processor systems

One of the processors is completely allocated to the *SoftPLC*. The *SoftPLC* cycle time is directly dependent on the execution time of OB1. A short OB1 will result in a great many executions during a given time period.

1.3 The *SoftPLC* Installation Program

The Install program performs all the required steps to setup the *SoftPLC* on the hard drive of your PC. Follow the instructions on the screen. You will be prompted to insert the **Key Disk** and to supply the path to the directory where you want to install the *SoftPLC*.

The file README.TXT will provide you with latest up to date information about the *SoftPLC* which may not be included in your manual at the date of shipment. Icons will be installed in the Program Manager to start the *SoftPLC* and the programs to enable the use of an external PC for online activities. Folders (directories) with examples that show you how to work with the *SoftPLC* are also installed.

The **Key Program** is used only to move or to alter the *SoftPLC* protection key.

If Windows is not currently running on your PC, start Windows NT 4.0 / Windows 2000.

1.3.1 Installing the *SoftPLC* under Windows NT 4.0 / Windows 2000

- Click ***Start, Run***.

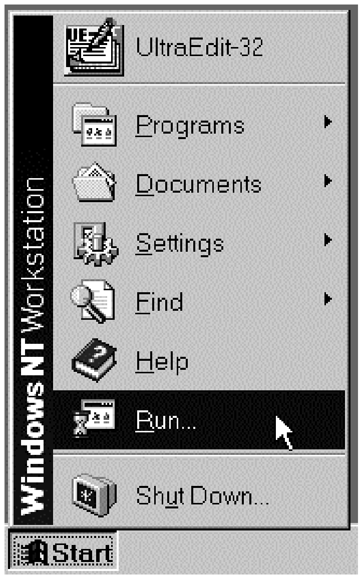


Figure 1

The **Run** dialog box opens.

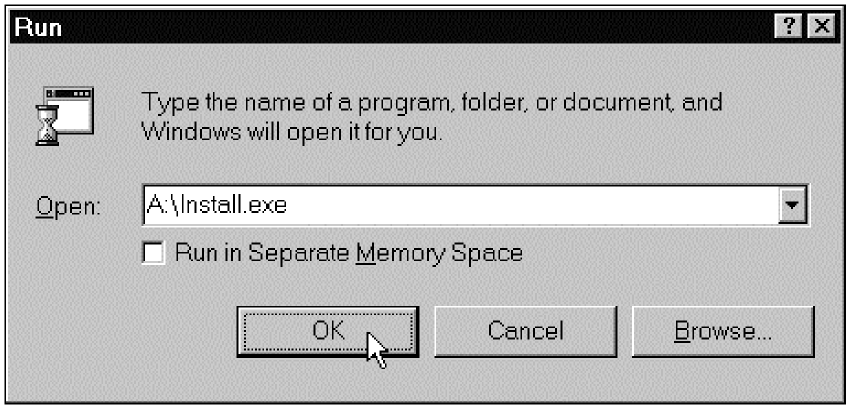

Run
Type the name of a program, folder, or document, and Windows will open it for you.
Open:
A:\Install.exe
Run in Separate Memory Space
OK
Cancel
Browse...

Figure 2

- Type **a:install** or **b:install** in the command line, this will depend on the designation of the drive where you inserted the program disk.

  Depending on the drive name designation where you inserted the *SoftPLC* installation disk # 1, with the button ***Browse***, you can search for the **install.exe** file.

- Click the ***OK*** button in the dialog box to start the install program.

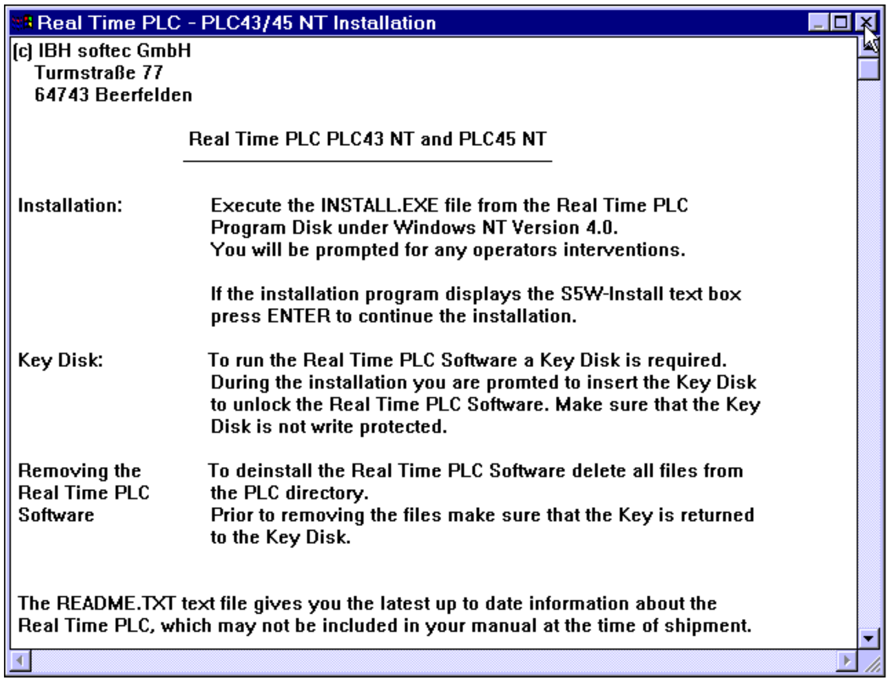
1.3.1.1 *SoftPLC* Start Up Text File

After starting the installation, a text file with important information is displayed (figure 3).

◆ Close the text file after reading it.

◆ To continue the installation, click the button in the upper right corner of the text field.

◆ Press **ALT+F4**.



Real Time PLC - PLC43/45 NT Installation

(c) IBH softec GmbH
Turmstraße 77
64743 Beerfelden

Real Time PLC PLC43 NT and PLC45 NT

Installation: Execute the INSTALL.EXE file from the Real Time PLC Program Disk under Windows NT Version 4.0. You will be prompted for any operators interventions.

If the installation program displays the S5W-Install text box press ENTER to continue the installation.

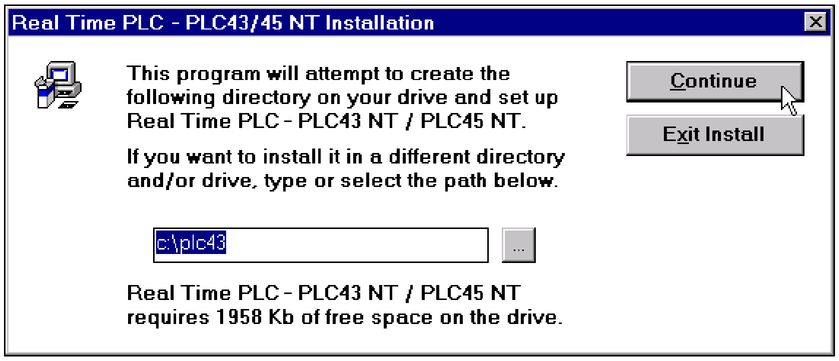
Key Disk: To run the Real Time PLC Software a Key Disk is required. During the installation you are promted to insert the Key Disk to unlock the Real Time PLC Software. Make sure that the Key Disk is not write protected.

Removing the Real Time PLC Software: To deinstall the Real Time PLC Software delete all files from the PLC directory. Prior to removing the files make sure that the Key is returned to the Key Disk.

The README.TXT text file gives you the latest up to date information about the Real Time PLC, which may not be included in your manual at the time of shipment.

Figure 3

1.3.1.2 *SoftPLC* Installation Menu



Real Time PLC - PLC43/45 NT Installation

This program will attempt to create the following directory on your drive and set up Real Time PLC - PLC43 NT / PLC45 NT.

If you want to install it in a different directory and/or drive, type or select the path below.

c:\plc43

Real Time PLC - PLC43 NT / PLC45 NT requires 1958 Kb of free space on the drive.

Figure 4

To install the *SoftPLC* software in a different folder, change the suggested drive and/or folder by entering a new drive and folder in the dialog box. You may open the dialog box to install the *SoftPLC* software in an existing path (folder and drive).

Attention:
For reasons of compatibility the default directory to install the *SoftPLC* is C:\PLC43 even when installing the PLC 45.

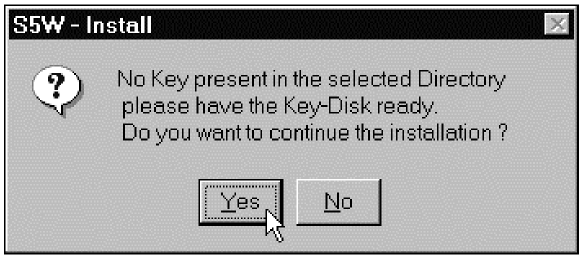
- ◆ Click ***Continue*** to continue with the installation program
- ◆ Press **È** to continue with the installation program.

The installation program creates the required folder and copies all the *SoftPLC* files to this folder. It also creates additional folders where you will find examples that will show you how to work with the *SoftPLC*. For more information see the **readme** files in the *SoftPLC* folder.

To abort the installation program the following procedure is required:

- ◆ Click the button ***Exit Install***.
- ◆ Select the button ***Exit Install***. With the **TAB** key, you may jump from one button or dialog box to the next.
- ◆ Confirm with the **È** key.

A dialog box opens to remind you that a **Key Disk** is required to unlock the *SoftPLC* software.

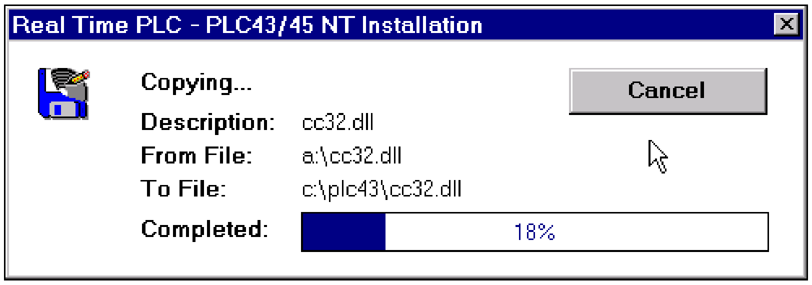

S5W - Install

No Key present in the selected Directory
please have the Key-Disk ready.
Do you want to continue the installation ?

Figure 5

- ◆ Click ***Yes*** to continue with the installation program.
- ◆ Press **È** to continue with the installation program.

The installation program indicates which file is being copied to the selected directory and the progress of the installation. With the indication of 100%, all the *SoftPLC* files are installed. The installation takes a few minutes.



Real Time PLC - PLC43/45 NT Installation

Copying...

Description: cc32.dll

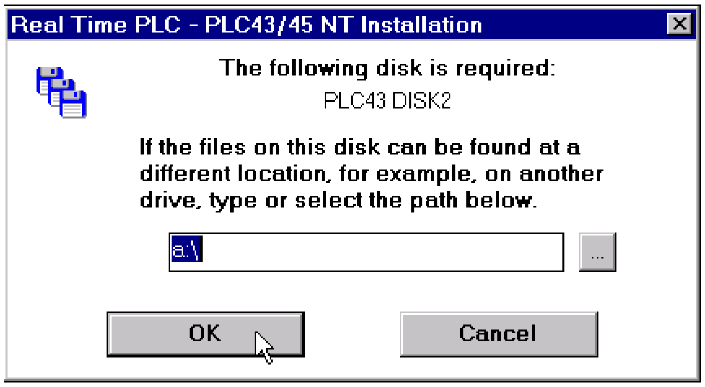
From File: a:\cc32.dll

To File: c:\plc43\cc32.dll

Completed: 18%

Figure 6

The *SoftPLC* software requires a second disk to complete the installation. Please insert the installation disk # 2 in the designated disk drive.



Real Time PLC - PLC43/45 NT Installation

The following disk is required:
PLC43 DISK2

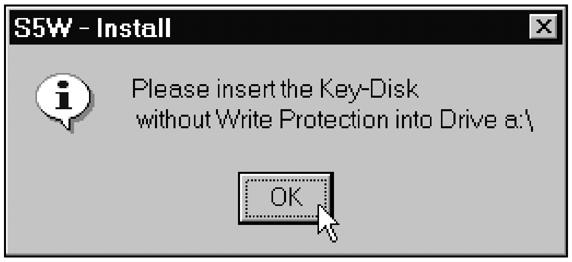
If the files on this disk can be found at a different location, for example, on another drive, type or select the path below.

a:\

Figure 7

Please confirm the insertion of the installation disk # 2 in the designated disk drive.

After copying all the required files, the installation program will prompt you to insert the **Key Disk** in the designated disk drive (figure 8).



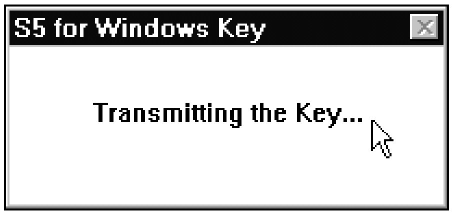
S5W - Install

Please insert the Key-Disk
without Write Protection into Drive a:\

Figure 8

Please confirm the insertion of the key disk in the designated disk drive. Make sure that the disk is not write protected.

The installation of the key is indicated with the following message.

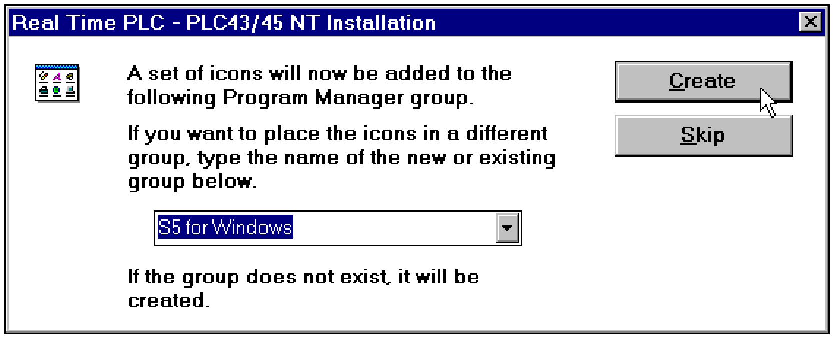


S5 for Windows Key

Transmitting the Key...

Figure 9

1.3.1.3 Adding the *SoftPLC* Icons


Real Time PLC - PLC43/45 NT Installation

A set of icons will now be added to the following Program Manager group.

If you want to place the icons in a different group, type the name of the new or existing group below.

S5 for Windows

If the group does not exist, it will be created.

Figure 10

The installation program will ask you if it should add a group to the Windows program manager. In this group the installation program will place the icon to start the *SoftPLC*.

Windows NT puts the icons into the ***Start***, ***Programs*** menu. By default the *SoftPLC* installation program will put the icons into the *S5 for Windows* group. Another group may be select by entering the new name into the dialog box.

- ◆ Click the ***Create*** button.
- ◆ Confirm the group name with the È key.

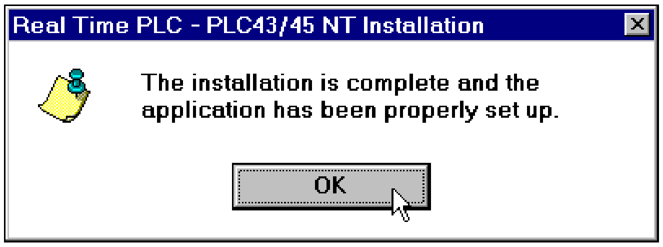
If you don't want to create a new group, skip this dialog box.

- ◆ Click the button ***Skip***.
- ◆ Select the button ***Skip*** and confirm with the È key.

1.3.1.4 Completing the Installation

The successful installation process ends with the message box shown below (figure 11). Confirm the complete installation.

In the next chapter, we will explain the steps required to start the *SoftPLC* for the first time.


Real Time PLC - PLC43/45 NT Installation

The installation is complete and the application has been properly set up.

Figure 11

WARNING:

You must recover the key from the *SoftPLC* directory to the key disk PRIOR to performing any one of the following tasks:

- **Deleting the directory of the *SoftPLC* or any system file*.***
- **Formatting the hard drive where the directory of the *SoftPLC* is located.**
- **Removing the hard drive where the directory of the *SoftPLC* is located from the PC for no further use.**

If you do NOT, you will lose the key required to unlock the *SoftPLC*.

For more information on the key disk see the appendix of this manual.

2 Starting the *SoftPLC* the first Time

The installation program added icons to the *S5 for Windows®* group to start the *SoftPLC*, to access the *SoftPLC* from an external operator's panel or PC (online mode), and to open the ***readme*** text file.

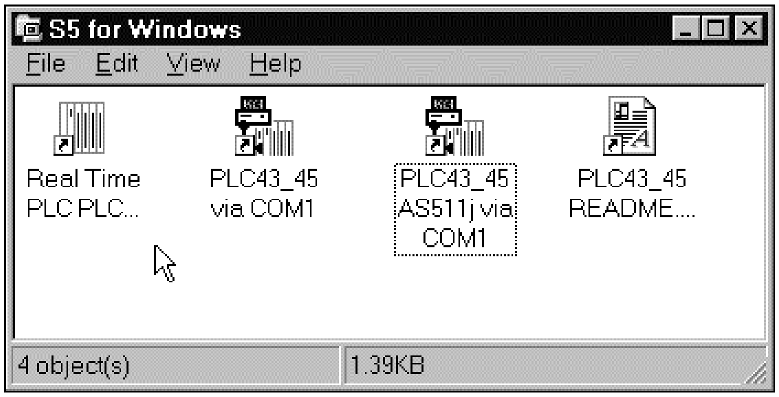


Figure 11

To read the most recent up to date information about the *SoftPLC* software, which may not be included in your manual at the time of shipment, double click the PLC43_45 Readme icon.

If the *SoftPLC* installation program has created the *S5 for Windows®* program group you can start the *SoftPLC* as followed:

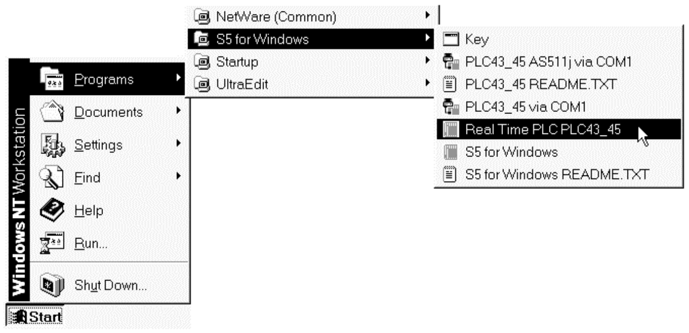
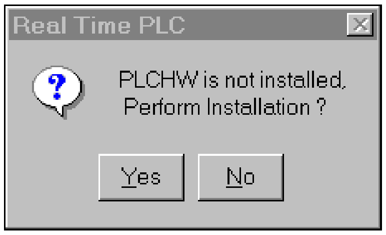


Figure 12

◆ Click **Start, Programs, *S5 for Windows®* , *SoftPLC* PLC43_45**

The following dialog box will open:


Real Time PLC

PLCHW is not installed.
Perform Installation ?

Yes No

Figure 13

The dialog box indicates that no interface driver for the *SoftPLC* communication has been installed.

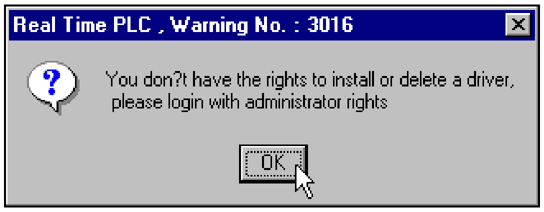
◆ Click **YES** to install the interface driver.

◆ Press **RETURN**.

Attention:

You must be logged on to the Windows NT / Windows 2000 system with **Administrator** rights.

If you are logged on to the system with normal user rights Windows will display the following warning:



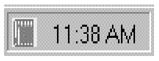
Real Time PLC , Warning No. : 3016

You don?t have the rights to install or delete a driver, please login with administrator rights

Figure 124

After confirming the warning you must restart Windows NT / Windows 2000 and you have to log on to the system with the administrator rights.

From now on the interface driver will be called automatically whenever you start Windows NT / Windows 2000.



To indicate that the *SoftPLC* software is running, an icon is displayed in the task bar next to the clock.

Note:

The color of the border surrounding the SoftPLC icon indicates the status of the SoftPLC:

Border color:

Yellow:	No PLC program for execution stored in the SoftPLC memory.
Red:	SoftPLC in the **STOP** mode.
Green:	SoftPLC in the **RUN** mode.

2.1 Direct access to the *SoftPLC* with *S5 for Windows®*

S5 for Windows® provides you with direct access to the *SoftPLC* to transfer PLC programs and to display the status. All of the *S5 for Windows®* online functions are directly available with the *SoftPLC*. The data transfer is extremely fast.

If you want to access the *SoftPLC* directly with *S5 for Windows®* you must copy the following files into the *S5 for Windows®* directory (usually C:\s5w):

S5 for Windows® Version 3.10 and higher:

- PLCMNT.DLL
- S5WPGI.DLL

S5 for Windows® Version 3.00 and higher but not 3.10 and higher:

- PLCMNT.DLL
- S5Wpgi30.DLL after copying rename the file into S5WPGI.DLL

If you own an *S5 for Windows®* version below 3.00 you need to update *S5 for Windows®*.

Attention:

The above mentioned files must be in the SoftPLC folder **and** in the *S5 for Windows®* folder.

Do not move the files copy the files.

2.1.1 Accessing the *SoftPLC* with *S5 for Windows®* from an external PC using a serial COM port

PLC43/45
via COM1

The *SoftPLC* installation sets up a program (PLCCOM.EXE) to access the *SoftPLC* via the COM1 port. The data transfer is set to 38 400 Baud. A null modem cable is required (see appendix) to connect the external PC with the PC running the *SoftPLC*. To select another COM port and/or another baud rate you must execute the interface driver program (PLCCOM.EXE) with the accompanying parameters.

COM1	PLCCOM.EXE 0
COM2	PLCCOM.EXE 1
COM3	PLCCOM.EXE 2
COM4	PLCCOM.EXE 3

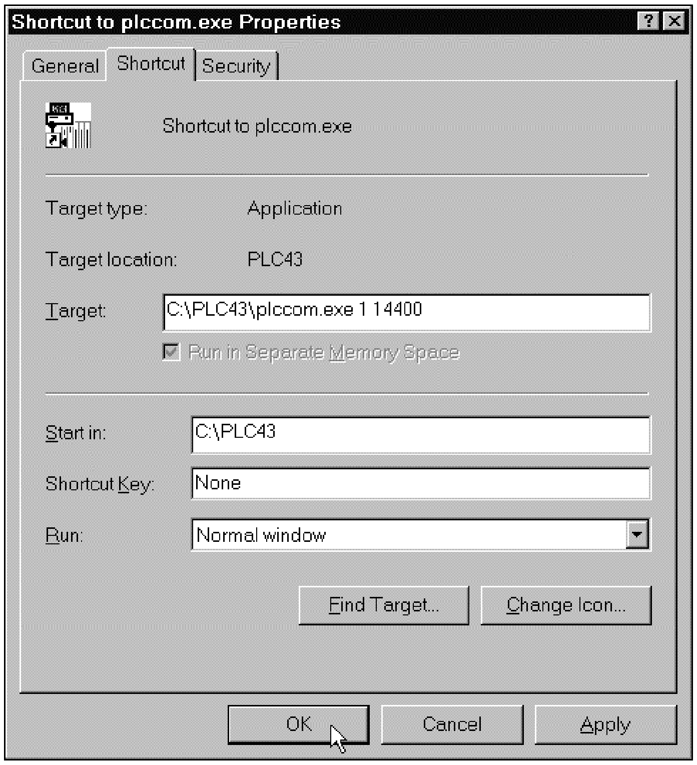
38 400 Baud	PLCCOM.EXE n 38800
19 200 Baud	PLCCOM.EXE n 19200
14 400 Baud	PLCCOM.EXE n 14400
9 600 Baud	PLCCOM.EXE n 9600

Example:

PLCCOM.EXE 1 14400 COM 2, Baudrate 14 400

If you want to add an icon to call a different serial port setting you may use the following procedure:

1. In the Windows NT / Windows 2000 Exploring Window mark the file PLCCOM.EXE (PLC43 folder).
2. Click the command ***Create Shortcut*** in the File menu. In the Windows NT / Windows 2000 Exploring Window the shortcut is displayed.



Shortcut to plccom.exe

Target type: Application

Target location: PLC43

Target: C:\PLC43\plccom.exe 1 14400

☑ Run in Separate Memory Space

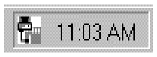
Start in: C:\PLC43

Shortcut Key: None

Run: Normal window

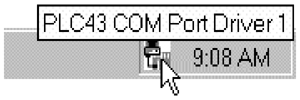
Figure 113

3. Click the command ***Properties*** in the File menu. The Shortcut dialog box is opened.
4. In the ***Target*** field (Shortcut tab) edit the required parameter behind the existing program call (plccom.exe).
5. Confirm the entered parameter with OK.



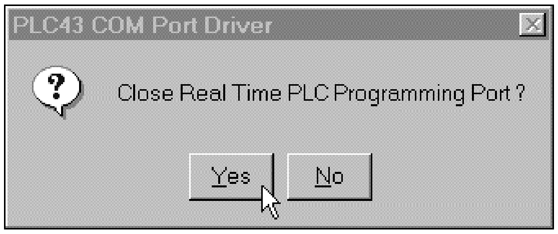
If the serial port is defined with the PLCCOM.EXE program, an icon is displayed in the task bar next to the clock.

The COM port in use is displayed when the mouse pointer is located at the PLCCOM.EXE program icon.



To change the baud rate or the COM port, the PLCCOM.EXE program must be closed and then reopened to implement the PLCCOM.EXE program with the new parameters. To close the serial port driver program click the PLCCOM.EXE program icon.

Figure 13



Close Real Time PLC Programming Port ?

2.1.2 Accessing the *SoftPLC* with an external Operators Panel

PLC43/45
AS511j via
COM1

The *SoftPLC* installation sets up an AS511 interface driver (AS511J.EXE) to access the *SoftPLC*, via the COM1 port, to read and write variables. The data transfer is set to 9600 Baud. A null modem cable is required (see appendix) to connect the external Operators Panel with the PC running the *SoftPLC*. To select another COM port and you must execute the interface driver program (AS511J.EXE) with the accompanying parameters.

COM1	AS511J.EXE 0
COM2	AS511J.EXE 1
COM3	AS511J.EXE 2
COM4	AS511J.EXE 3

Example:

AS511J.EXE 1 COM 2 Port

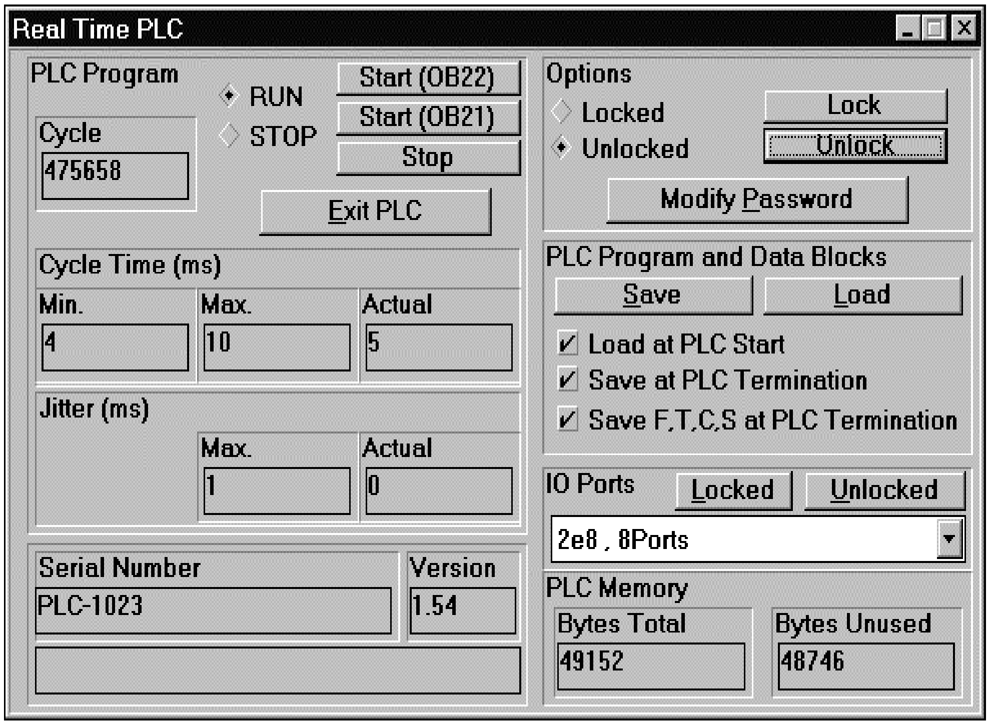
An AS511 – Driver providing all the programming system functions may be ordered separately. With this driver you may access the *SoftPLC* using a Siemens programming system.

3 *SoftPLC* Dialog Box

11:38 AM

After starting the *SoftPLC* (see chapter 2) you can open the SoftPLC dialog box by clicking the *SoftPLC* icon in the status bar.

3.1 SoftPLC Dialog Box Structure



Real Time PLC

PLC Program: ◆ RUN ◇ STOP — Start (OB22), Start (OB21), Stop, Exit PLC

Cycle: 475658

Cycle Time (ms): Min. 4, Max. 10, Actual 5

Jitter (ms): Max. 1, Actual 0

Serial Number: PLC-1023 — Version: 1.54

Options: ◇ Locked ◆ Unlocked — Lock, Unlock, Modify Password

PLC Program and Data Blocks: Save, Load

- ✓ Load at PLC Start
- ✓ Save at PLC Termination
- ✓ Save F,T,C,S at PLC Termination

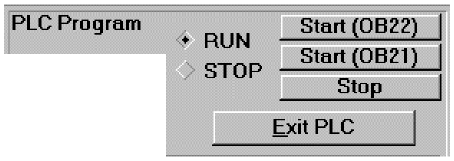
IO Ports: Locked, Unlocked — 2e8 , 8Ports

PLC Memory: Bytes Total 49152 — Bytes Unused 48746

Figure 13

The *SoftPLC* timing is displayed and constantly updated.

3.1.1 PLC Program



PLC Program: ◆ RUN ◇ STOP — Start (OB22), Start (OB21), Stop, Exit PLC

In the PLC Program field the status of the *SoftPLC* **"CPU"** is displayed:

◆ RUN — The PLC program is executed in the *SoftPLC* **"CPU"** (**RUN** mode)

◆ STOP — The *SoftPLC* **"CPU"** is in its **STOP** mode. The PLC program is not executed.

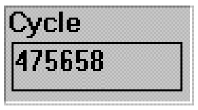
Start (OB22) — Marking the **Start (OB 22)** button will put the *SoftPLC* **"CPU"** is in its **RUN** mode. First, the Organization Block OB 22 is executed and then the PLC program will start to execute OB 1 and cycle. This program start is equal to the start of a hardware PLC when power is restored.

Marking the **Start (OB 21)** button will put the *SoftPLC* **“CPU”** is in its **RUN** mode. First, the Organization Block OB 21 is executed and then the PLC program will start to execute OB 1 and cycle. This program start is equal to the start of a hardware PLC when restarting the CPU with the RUN / STOP switch.

Marking the **Stop** button will put the *SoftPLC* **“CPU”** is in its **STOP** mode. The execution of the cycling PLC program is terminated. The state of the Process Images, Flags, Timer, and Counters are saved with their actual current status. The watchdog of the interface board must put the output signals into their ***OFF*** stage.

Marking the **Exit PLC** button will terminate the *SoftPLC*. You can restart the *SoftPLC* software any time (see chapter 2).

3.1.2 Cycle



Cycle

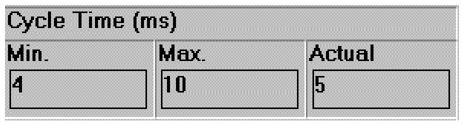
475658

The number of the cyclic PLC program executes is counted and displayed. The cycle counter is halted with the **Stop** button. Marking one of the **Start (OB2x)** buttons resets the cycle counter.
The number of program cycles (OB1 cycles) can be read by the PLC program.

• RS 20:	low value word.
• RS 21:	high value word.

3.1.3 Cycle Time

The time of the program execution is constantly measured and displayed. The minimum, maximum, and the actual current cycle times of the PLC program being executed are displayed.



Cycle Time (ms)

Min.	Max.	Actual
4	10	5

The cycle time of a PLC program may vary greatly depending of the PLC program structure.

At the end of a cycle, the *SoftPLC* saves the result of the time measurement. The time measured is the time from the call of OB 1 until the next OB 1 call.

The cycle times (OB 1 cycle time) can be read by the PLC program.

• RS 121:	Currently actual cycle time.
• RS 122:	Maximum cycle time.
• RS 123:	Minimum cycle time.

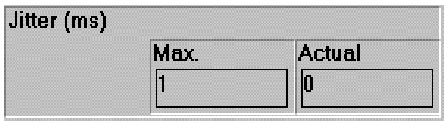
3.1.4 Jitter

The ***Jitter*** is the differences in the cycle time of the *SoftPLC* caused by the system itself (see chapter 4.1).

If a pulse is outputted from a timer OB (e.g. OB 10, every 10 ms) and measured, differences may be found. The jitter is the variation in time measured relative to the previous measured time.

You will see jitter with hardware and software PLC's.

With software PLCs the jitter may be caused by programmed interruption within the PLC program and also could occur due to hard disk access, network access, etc (system activities).



Max.	Actual
1	0

The ***Jitter*** of the *SoftPLC* is constantly measured. The maximum jitter and the currently actual jitter are displayed.
The jitter value must be added to the reaction time of the *SoftPLC*.

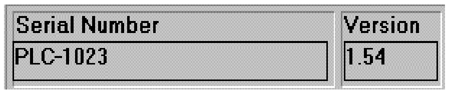
This jitter value may be read from the *SoftPLC* via the PLC program:

- RS 24: Maximum jitter in ms.
- RS 25: Actual (momentary) jitter in ms.

The hardware of the PC executing the *SoftPLC* should be configured to minimize the jitter of the *SoftPLC* (see chapter 1.1).

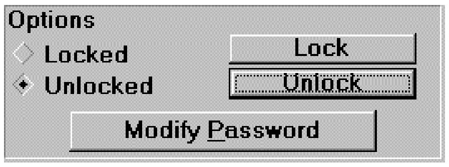
3.1.5 Serial Number and Version

The *SoftPLC* software serial number and its version number are displayed.



Serial Number	Version
PLC-1023	1.54

3.1.6 Options

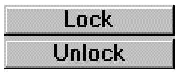


Options: Locked [✓ Unlocked]

The *SoftPLC* options to be set with the *SoftPLC* dialog box are interlocked with a password. The settings in the PLC Program and Data Blocks (see chapter 3.1.7) and the I/O port settings (see chapter 3.1.8) can only be altered when the options are unlocked with the correct password.

The marked field **Locked** indicates that non of the settings in the *SoftPLC* dialog box may be altered prior to unlocking the options with a password.

The marked field **Unlocked** indicates that the option may be altered any time without a password.



The locking and unlocking of the options is done with the buttons **Lock** and **Unlock**. These buttons open the text box to enter the password.

If **Locked** is marked, none of the *SoftPLC* dialog box settings can be altered. The operation of the buttons Start, Stop, and Exit PLC is also disabled. The locking option gives you the ability to inhibit any operators invention after setup. To change the *SoftPLC* dialog box settings, a valid password must be entered prior to the intervention. The information in the dialog box is always valid.

To start the *SoftPLC* automatically when booting the PC, the *SoftPLC* software (PLC.EXE) should be connected with the auto start routine from Windows NT. If the AS511 driver and/or communication via a serial port is needed, the corresponding files (AS511J.EXE, PLCCOM.EXE), with the required parameters (see chapter 2.1.1, 2.1.2), may also be connected with the auto start routine.

◆ Locked
◆ Unlocked

If one of these buttons is pressed, the **Enter Password** dialog box is opened. Please enter the valid password and confirm with **OK** button.

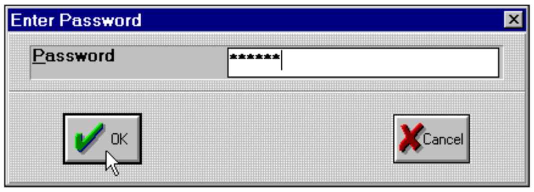

Enter Password
Password ******

Figure 14

Modify Password

You may also change the password by pressing the **Modify Password** button. Prior to activating the *SoftPLC* lock protection, you have to define the password to turn the protection on and off. The password may have up to 27 characters. The password may be changed any time as long as you know the old password. As a default no password is assigned.

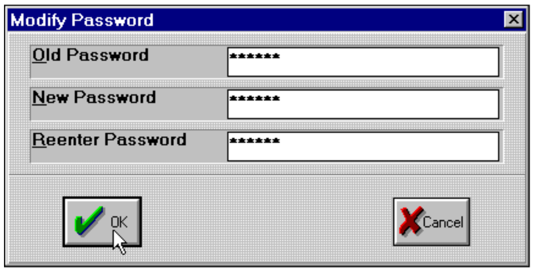

Modify Password
Old Password ******
New Password ******
Reenter Password ******

Figure 145

In the text field **Old Password** enter the password you defined the last time. If no password was yet assigned (default setting) do not enter a password.

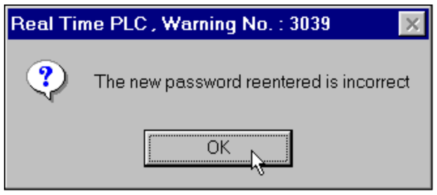
In the text field **New Password** enter the password you want to use to turn the *SoftPLC* lock protection on. The same password is needed to turn the PLC lock protection off.

To avoid entering a wrong password due to typing errors you have to type the new password again in the text field **Reenter Password**.

The password you entered is not displayed in the text fields. Each star represents one (1) typed character.

- To confirm the password entry, activate the **OK** button. The *SoftPLC* lock protection may now be turned on and of with the defined password.

If the reentered password and the new password do not match a warning will be displayed and the new password is ignored.


Real Time PLC , Warning No. : 3039

The new password reentered is incorrect

Figure 16

- Confirm the warning.

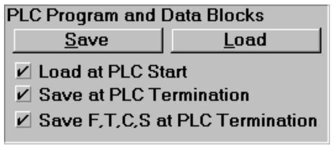
The modify password dialog box is still open.

We recommend that you reenter the old password, the new password and to type the new password in the reenter password text field.

In case you lost the password you have to reinstall the *SoftPLC*. To do so make sure the key is transferred back to the key disk prior to reinstalling the *SoftPLC* (see appendix on how to handle the key disk).

3.1.7 PLC Program and Data Blocks

Buttons are provided to control the loading and termination of the PLC program.

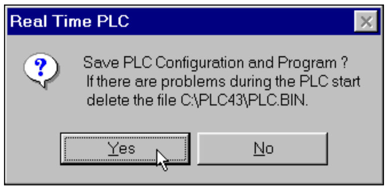

PLC Program and Data Blocks

✓ Load at PLC Start
✓ Save at PLC Termination
✓ Save F,T,C,S at PLC Termination

Figure 17

Activating the **Save** button will save the PLC program currently stored in the *SoftPLC* to the hard disk in the file **C:\PLC43\PLC.BIN**. If the check box **Save F, T, C, S at PLC Termination** is activated the Flags (F), Timers (T), Counters (C), and the special Flags (S) are also saved in the **PLC.BIN** file.

The *SoftPLC* will display the following warning prior to the actual saving on disk:


Real Time PLC

Save PLC Configuration and Program ?
If there are problems during the PLC start
delete the file C:\PLC43\PLC.BIN.

Figure 18

If you encounter problems when starting the *SoftPLC* software the next time delete the **C:\PLC43\PLC.BIN** file. Problems may occur if a corrupted PLC program was saved or the actual saving process was interrupted. In such a case you must reload the PLC program from the programming system environment. The Flags (F), Timers (T), Counters (C), and the special Flags (S) will then be in their initial conditions.

The PLC program, stored on the hard disk in the **C:\PLC43\PLC.BIN** file, will be transferred into the *SoftPLC*. If the check box **Save F, T, C, S at PLC Termination** was activated prior to the last save action the Flags (F), Timers (T), Counters (C), and the special Flags (S) are reset to the condition they had at the save process. The **Jitter** display is reset.

✓ Load at PLC Start If the **Load at PLC Start** check box is activated, the PLC program stored on the hard disk in the **C:\PLC43\PLC.BIN** file will be transferred into the *SoftPLC* during the *SoftPLC* start up procedure (PC boot up). The start of the *SoftPLC* is exactly the same as if starting the hardware PLC after a power failure (the power to the hardware PLC was switched off). First, the Organization Block OB22 will be executed. Then the cyclical execution of the Organization Block OB1 will start. If the Flags (F), Timers (T), Counters (C), and the special Flags (S) have been saved in the **C:\PLC43\PLC.BIN** file, the saved stage will be transferred into the *SoftPLC*.

✓ Save at PLC Termination If the **Save at PLC Termination** check box is activated, the PLC program resident in the *SoftPLC* is saved in the **PLC.BIN** file on the hard disk when the *SoftPLC* software is terminated. In addition, the PLC program is saved in the **PLC.BIN** file when Windows NT is terminated with its normal procedure while the *SoftPLC* is running.

✓ Save F,T,C,S at PLC Termination If the **Save F, T, C, S at PLC Termination** check box is activated, the current condition of the Flags (F), Timers (T), Counters (C), and the special Flags (S) are saved in the **PLC.BIN** file along with the PLC program currently resident in the *SoftPLC*, when the *SoftPLC* software is terminated. In addition, the PLC program is saved in the **PLC.BIN** file when Windows NT is terminated with its normal procedure while the *SoftPLC* is running.

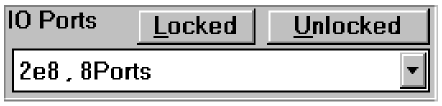
With the explanation of the use of these check boxes and an Uninterruptible Power Supply (UPS) it is possible to configure the *SoftPLC* to act in the same way as a hardware PLC in case of an power failure (retentive behavior).

With an installed UPS and the support of Windows NT, the PLC program resident in the *SoftPLC* and the Flags (F), Timers (T), Counters (C), and the special Flags (S) with their current conditions are saved. A signal from the UPS (via a COM port) starts the power down procedure in the case of an power failure. The battery of the UPS will ensure that the PC has power until the power down process is completed.

If the power returns, the *SoftPLC* program may start with the same conditions of the Flags (F), Timers (T), Counters (C), and the special Flags (S) that were saved at power down.

3.1.8 I/O Ports

I/O ports can be directly linked to the *SoftPLC* for communication.

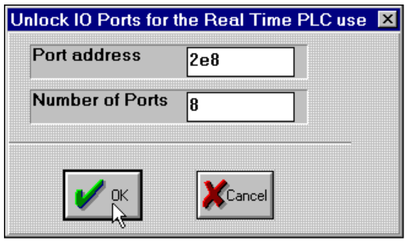


2e8 , 8Ports

Figure 19

Unlocked The button **Unlocked** opens the dialog box **Unlock IO Ports for the *SoftPLC* use**.

The port address must be entered into the text field in hexadecimal format. The number of ports to be used, starting at the entered address, must also be entered in the **Number of Ports** text field.

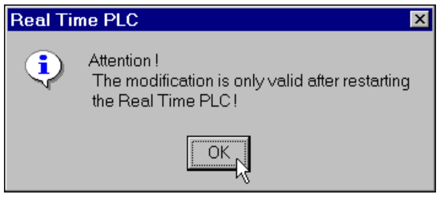


Port address: 2e8
Number of Ports: 8

Figure 150

Activate the OK button to confirm the input.

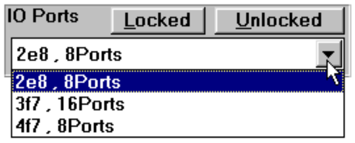
The *SoftPLC* will display the following warning to indicate, that the unlocking of the ports will only be active after restarting the *SoftPLC*.



Attention !
The modification is only valid after restarting the Real Time PLC !

Figure 21

The addresses and the number of all unlocked ports are displayed in the pull down list.



2e8 , 8Ports
2e8 , 8Ports
3f7 , 16Ports
4f7 , 8Ports

Figure 22

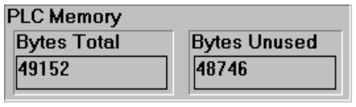
Attention:

The security of Windows NT requires, that the IO ports must be unlocked prior to accessing the ports with the PLC program.

If the PLC program tries to access an IO port which is **NOT** unlocked Windows NT will display a protection warning and terminates the *SoftPLC* software.

3.1.9 PLC Memory

The total memory available for the *SoftPLC* program and the memory used by the PLC program are displayed.



Bytes Total	Bytes Unused
49152	48746

Figure 216

The **PLC43 NT** has a PLC program memory of 48 Kbyte (= 49,125 byte).

The **PLC45 NT** has a PLC program memory of 720 Kbyte (= 737,280 byte).

Special instructions are implemented to enable the PLC program to read information about the size and use of the PLC program memory. These instructions are very useful when implementing special functions with the Graph® 5 diagnostics.

With PLC43:

- RS 33 = PLC program memory already used
- RS 36 = Start address of the internal RAM (PLC program memory)
- RS 37 = End address of the internal RAM (PLC program memory)

With PLC45:

- RS 36 - 37 = PLC program memory already used
- RS 32 - 33 = Start address of the internal RAM (PLC program memory)
- RS 34 - 35 = End address of the internal RAM (PLC program memory)

4 *SoftPLC* PLC43

In the following chapter the differences between the *SoftPLC*43 and the Siemens CPU943 are described. Also, all the instructions the *SoftPLC*43 can execute (operations list, special Instructions) and the memory allocation are listed.

4.1 Additional Features in the *SoftPLC*43

The following additional features are integrated in the *SoftPLC*43.

- ◆ Additional Special Instructions DO RS 10 .. DO RS 27
- ◆ The PID Block OB251 is extremely fast. About 100 μs with a 486-66 PC.
- ◆ Double Word instructions.

4.2 Differences between the *SoftPLC*43 and the Siemens CPU 943

- ◆ ***"Clear All Blocks"*** from the PLC Block List executes an **Overall Reset.**
- ◆ An endless loop within a **Disable Interrupt** will hang up the PC program.
- ◆ If an Enable Interrupt is not programmed, it is automatically inserted at the end of the OB.

4.3 Functions not Incorporated in the *SoftPLC*43

Due to the physical differences between a hardware PLC and a PC executing a Windows application, not all of the functions in a hardware PLC can be integrated in the *SoftPLC*43. However, a program written to be executed on a PLC U115 should run on the *SoftPLC* with a minimum of modifications.

The following functions are not part of the *SoftPLC*:

- ◆ Flags are non retentive (also see chapter 3.1.7)
- ◆ Only the Following System Data Words have their integrated functions.

- SD 8..12	(Integral Clock)
- SD 96	(Scan Monitoring Time)

- SD 97..100 (Time controlled program scanning for OB10..OB13)
- SD 101 (Interrupt generated by internal timers for OB6)
- SD 121..123 (Scan Time)
- SD 203..238 (Interrupt Stack)

All other System Data Words are not used in the *SoftPLC*.

- ◆ Timer OB's and Interrupt OB's are not interrupted by Windows. Therefore these OB's may not have a long execution time (<1ms).
- ◆ The OB6 can not interrupt the Timer.
- ◆ The Data Handling Blocks FB244..FB249 are not available.
- ◆ The integrated Clock has no alarm function. The operating hours counter and the input of a correction value to compensate for clock inaccuracy are not implemented. The System Data Area of the *SoftPLC* is equal to the System Data Area of the CPU 943. The above mentioned functions have no meaning in the *SoftPLC*.
- ◆ No setting of parameters for internal functions via DB1.

4.4 Operations of the PLC43

The execution times specified are guideline values for the instructions. The actual execution time of an instruction depends on the processor used and its speed. To calculate the execution time of a PLC Program, the time slice setting of the *SoftPLC* must be taken into consideration.

In a **single processor system** the time slice is set to 50%, the PC is executing the *SoftPLC* 50% of its time and the other 50% is used by Windows. You have to multiply the calculated execution time of a PLC program by two.

In a **dual processor system,** one CPU is completely allocated to the *SoftPLC*. No time slice has to be taken into consideration.

Having short PLC programs, you must consider that the *SoftPLC* is called with a fixed time pulse (2ms). The time pulse will start the execution of OB1. With one time pulse call, the OB1 will be executed only once.

4.5 *SoftPLC*43 Special Instructions

The following special instructions are implemented in the *SoftPLC*43 (Windows NT version):

- ◆ DO RS 12 : Assigning Peripheral Bytes to a Hardware Ports
- ◆ DO RS 17 : Read from a Hardware Port (single bytes direct)
- ◆ DO RS 30 : Read from a Hardware Port (single words direct)

- DO RS 18 : Write into a Hardware Port (single bytes direct)
- DO RS 31 : Write into a Hardware Port (single words direct)
- DO RS 20 : Read from a Memory Location and store the Contents in the Low Byte of ACCU 1
- DO RS 21 : Write the Low Byte from ACCU2 into a Memory Location
- DO RS 22 : Load Data into multiple Memory Locations of the 115U Memory
- DO RS 23 : Store Data from multiple Memory Locations of the 115U Memory
- DO RS 28 : Call a C Program procedure

4.5.1 DO RS 12 : Assigning Peripheral Bytes to a Hardware Ports

Call:

A DB <AssignmentDB>

L KF <DW Address of the 1st Assignment (n)>

L KF <Number of Assignments>

DO RS 12

AssignmentDB Structure:

DRn =Peripheral Byte No. (0..255)

DLn = 0: read + write

1: read only

2: write only

DWn+1 = Hardware Port address

DWn+2 = next Assignment

4.5.2 DO RS 17 : Read from a Hardware Port (single bytes direct)

Call:

L KH <Hardware Port Address>

DO RS 17

After the Call:

ACCU1: Data Byte read

4.5.3 DO RS 18 : Write into a Hardware Port (single bytes direct)

Call:

L <Data Byte to be written to the Port>

L KH <Hardware Port Address>

DO RS 18

4.5.4 DO RS 20 : Read a Memory Location and store the Contents in the Low Byte of ACCU 1

Call:

L DH <Source address>(Linear 32 Bit address)

Example: 000D0000 → D000:0

DO RS 20

After the Call:

ACCU1 (low Byte) : Data Byte read

4.5.5 DO RS 21 : Write the Low Byte from ACCU2 into a Memory Location

Call:

L <Data Word>

L DH <Destination address>(Linear 32-Bit address)

Example: 000D0000 → D000:0

DO RS 21

4.5.6 DO RS 22 : Load Data into multiple Memory Locations of the 115U Memory

Call:

L DH <Source address>(Linear 32-Bit Address)

Example: 000D0000 → D000:0

L DH <Length, Address at the 115U Memory>

DO RS 22

The address for the 115U is built from the lower 5 hexadecimal digits. The upper 3 hexadecimal digits define the length.

The *SoftPLC*43 emulates the Memory of the PLC 115U (CPU 943). For more details see chapter 4.7 of this manual.

Example:

```
DH 000D0000
DH FA011000
DO RS 22
```

Transfer 4000 Bytes starting from address D000:0000 to the 115U Memory starting at the memory location 11000h.

4.5.7 DO RS 23 : Store Data from multiple Memory Locations of the 115U Memory

Call:

L DH	<Length, Address at the 115U Memory>
L DH	<Destination address>(Linear 32-Bit Address)

Example: 000D0000 → D000:0

DO RS 23

4.5.8 DO RS 28 : Call a C program procedure

Call:

L KH	<C procedure number>

DO RS 28

The *SoftPLC* software can handle up to 256 C procedures. In the directory **\CB123** you will find an example of a C procedure call. To compile the C program the Microsoft Visual C++ version 4.x is required.

The C programs are linked to the *SoftPLC* software whenever the *SoftPLC* software starts to be executed. With the start of the *SoftPLC* software the *SoftPLC* directory is searched for the DLL's **CB000.DLL** up to **CB255.DLL**. The DLL's recognized are linked into the *SoftPLC* software. If you compiled a C program again, you must restart the *SoftPLC* software to link the new compiled C program into the *SoftPLC* software.

Each C procedure exports a C function:

```
Extern "C" void C_PROGRAMM (TCPU * cpu)
```

With the parameter "cpu" from the C function you have access to the ACCU1, ACCU2, and the 115 memory structure.

Attention:

The C procedure is executed with the highest Windows priority. The execution time of the C procedure directly influences the execution time of the *SoftPLC* software and therefore takes influences of the real time capability of the *SoftPLC*. Please consider the following rules:

- Do not call any Windows function
- Do not call a function requiring operators intervention
- Call only functions having a defined execution time

4.5.9 DO RS 30 : Read from a Hardware Port (single words direct)

Call:

L KH <Hardware Port Address>

DO RS 30

After the Call:

ACCU1: Data word read

4.5.10 DO RS 31 : Write into a Hardware Port (single words direct)

Call:

L <Data Byte to be written to the Port>

L KH <Hardware Port Address>

DO RS 18

4.6 *SoftPLC*43 Standard Instructions

The following standard instructions are implemented in the *SoftPLC*43 (Windows NT version):

4.6.1 Binary Scan Operations

4.6.1.1 Scan Operand for "1" and combine with RLO through logic AND

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
A	I 0.0 .. 127.7	0,9	0,4	Scan Input
A	Q 0.0..127.7	0,9	0,4	Scan Output
A	F 0.0..255.7	0,9	0,4	Scan Flag
A	S 0.0..4095.7	1,4	0,6	Scan S Flag
A	D 0.0..255.15	1,6	0,8	Scan Data Word
A	T 0..255	1,3	0,6	Scan Timer
A	C 0..255	1,1	0,6	Scan Counter
A=	<Parameter>	1,9	0,9	Scan FB Parameter (Type : BI)

4.6.1.2 Scan Operand for "0" and combine with RLO through logic AND

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
AN	I 0.0 .. 127.7	0,9	0,4	Scan Input
AN	Q 0.0..127.7	0,9	0,4	Scan Output
AN	F 0.0..255.7	0,9	0,4	Scan Flag
AN	S 0.0..4095.7	1,4	0,6	Scan S Flag
AN	D 0.0..255.15	1,6	0,8	Scan Data Word
AN	T 0..255	1,3	0,6	Scan Timer
AN	C 0..255	1,1	0,6	Scan Counter
AN=	<Parameter>	1,9	0,9	Scan FB Parameter (Type : BI)

4.6.1.3 Scan Operand for "1" and combine with RLO through logic OR

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
O	I 0.0 .. 127.7	0,9	0,4	Scan Input
O	Q 0.0..127.7	0,9	0,4	Scan Output
O	F 0.0..255.7	0,9	0,4	Scan Flag
O	S 0.0..4095.7	1,4	0,6	Scan S Flag
O	D 0.0..255.15	1,6	0,8	Scan Data Word
O	T 0..255	1,3	0,6	Scan Timer
O	C 0..255	1,1	0,6	Scan Counter
O=	<Parameter>	1,9	0,9	Scan FB Parameter (Type : BI)

4.6.1.4 Scan Operand for "0" and combine with RLO through logic OR

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
ON	I 0.0 .. 127.7	0,9	0,4	Scan Input
ON	Q 0.0..127.7	0,9	0,4	Scan Output
ON	F 0.0..255.7	0,9	0,4	Scan Flag
ON	S 0.0..4095.7	1,4	0,6	Scan S Flag
ON	D 0.0..255.15	1,6	0,8	Scan Data Word
ON	T 0..255	1,3	0,6	Scan Timer
ON	C 0..255	1,1	0,6	Scan Counter
ON=	<Parameter>	1,9	0,9	Scan FB Parameter (Type : BI)

4.6.1.5 Other Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
O	-	2,3	1,1	Combine AND operations through logic OR
A(	-	2,3	1,1	Combine expressions enclosed in parentheses through logic AND (8 levels possible)
O(	-	2,3	1,1	Combine expressions enclosed in parentheses through logic OR (8 levels possible)
)	-	2,3	1,1	Closing parenthesis
AW	-	0,5	0,2	ACCU1=ACCU1 AND ACCU2 (16 Bit)
OW	-	0,5	0,2	ACCU1=ACCU1 OR ACCU2 (16 Bit)
XOW	-	0,5	0,2	ACCU1=ACCU1 XOR ACCU2 (16 Bit)

4.6.2 SET / RESET Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
S	I 0.0 .. 127.7	1,0	0,6	Set Input
S	Q 0.0..127.7	1,0	0,6	Set Output
S	M 0.0..255.7	1,0	0,6	Set Flag
S	S 0.0..4095.7	1,4	0,8	Set S Flag
S	D 0.0..255.15	1,9	0,9	Set Data Word
S=	<Parameter>	1,9	0,9	Set FB Parameter (Type : BI)
R	I 0.0 .. 127.7	1,0	0,6	Reset Input
R	Q 0.0..127.7	1,0	0,6	Reset Output
R	M 0.0..255.7	1,0	0,6	Reset Flag
R	S 0.0..4095.7	1,4	0,8	Reset S Flag
R	D 0.0..255.15	1,9	0,9	Reset Data Word
RB=	<Parameter>	1,9	0,9	Reset FB Parameter (Type : BI)
=	I 0.0 .. 127.7	1,0	0,5	Assign Input
=	Q 0.0..127.7	1,0	0,5	Assign Output
=	M 0.0..255.7	1,0	0,5	Assign Flag
=	S 0.0..4095.7	1,4	0,7	Assign S Flag
=	D 0.0..255.15	1,9	0,9	Assign Data Word
==	<Parameter>	1,9	0,9	Assign FB Parameter (Type : BI)

4.6.3 Load Operations

4.6.3.1 Load Variable

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
L	IB 0 .. 127	0,5	0,2	Load Input Byte
L	QB 0..127	0,5	0,2	Load Output Byte
L	FB 0..255	0,5	0,2	Load Flag Byte
L	SY 0..4095	0,7	0,3	Load S Flag Byte
L	DL 0..255	1,5	0,8	Load Data Byte (left Byte)
L	DR 0..255	1,5	0,8	Load Data Byte (right Byte)
L	PY 0 ..255			Load Peripheral Byte
L	IW 0 .. 126	0,6	0,25	Load Input Word
L	QW 0..126	0,6	0,25	Load Output Word
L	FW 0..254	0,6	0,25	Load Flag Word
L	SW 0..4094	0,8	0,3	Load S Flag Word
L	DW 0..254	1,6	0,8	Load Data Word
L	PW 0 ..255			Load Peripheral Word
L	ID 0 .. 124	0,7	0,3	Load Input Double Word
L	QD 0..124	0,7	0,3	Load Output Double Word
L	FD 0..252	0,7	0,3	Load Flag Double Word
L	SD 0..4092	0,9	0,4	Load S Flag Double Word
L	DD 0..252	1,8	0,9	Load Data Double Word
L	T 0 ..255	0,7	0,3	Load Time (Binary Code)
L	C 0..255	0,7	0,3	Load Count (Binary Code)
LD	T 0 ..255	2,5	1,5	Load Time (BCD Code)
LD	C 0..255	2,5	1,5	Load Count (BCD Code)
L	RS 0 ..255	0,5	0,2	Load a System Data Word

4.6.3.1 Load Variable (continued)

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
LIR	0 (ACCU1) 2 (ACCU2)	0,7	0,3	Load the contents of a Memory Word (addressed by ACCU1) indirectly into the Register (0: ACCU 1; 2: ACCU 2)
LDI	A1 (ACCU1) A2 (ACCU2)	0,7	0,3	Load the contents of a Memory Word (addressed by 32 Bit ACCU1) indirectly into the Register (A1 = ACCU 1; A2 = ACCU 2)
L=	<Parameter>	1,5	0,65	Load the value of the formal operand into ACCU 1 (parameter type: BY, W, DW)
LC=	<Parameter>	4,0	2,0	Load the value of the formal operand in BCD code into ACCU 1 (parameter type: T,Z)
LW=	<Parameter>	0,8	0,35	Load a formal operand bit pattern into ACCU 1 (parameter type : D; Data type: KF,KH,KM,KY,KC,KT,KZ)

4.6.3.2 Load constant

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
L	KB 0 .. 255	0,4	0,2	Load a constant into ACCU 1 (Bit 0..7) and 0 into Bit 8..15
L	KS ‘AA’	0,4	0,2	Load a constant (2 characters in ASCII format) into ACCU 1
L	KF -32768 .. +32767	0,4	0,2	Load a constant (fixed point number) into ACCU1
L	KH 0..FFFF	0,4	0,2	Load a constant (hexadecimal) into ACCU 1
L	DH 0 .. FFFFFFFF	0,7	0,3	Load a constant (hexadecimal double word)
L	KM <16 Bit pattern>	0,4	0,2	Load a constant (bit pattern) into ACCU 1
L	KT 0.0 .. 999.3	0,4	0,2	Load a constant (Timer value) into ACCU 1 (BCD coded)
L	KY 0 .. 255, 0..255	0,4	0,2	Load a constant (2 byte number) into ACCU 1
L	KC 0 .. 999	0,4	0,2	Load a constant (Counter value) into ACCU 1 (BCD coded)

4.6.4 Transfer Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
T	IB 0 .. 127	0,5	0,2	Transfer the contents of ACCU 1 to an Input Byte
T	QB 0..127	0,5	0,2	Transfer the contents of ACCU 1 to an Output Byte
T	FB 0..255	0,5	0,2	Transfer the contents of ACCU 1 to a Flag Byte
T	SY 0..4095	0,7	0,3	Transfer the contents of ACCU 1 to a Special Flag Byte
T	DL 0..255	1,5	0,8	Transfer the contents of ACCU 1 to a Data Word (left Byte)
T	DR 0..255	1,5	0,8	Transfer the contents of ACCU 1 to a Data Word (right Byte)
T	PY 0 ..255			Transfer the contents of ACCU 1 to a Peripheral Byte
T	IW 0 .. 126	0,5	0,3	Transfer the contents of ACCU 1 to an Input Word
T	QW 0..126	0,5	0,3	Transfer the contents of ACCU 1 to an Output Word
T	FW 0..254	0,5	0,3	Transfer the contents of ACCU 1 to a Flag Word
T	SW 0..4094	0,8	0,4	Transfer the contents of ACCU 1 to a Special Flag Word
T	DW 0..254	1,6	0,8	Transfer the contents of ACCU 1 to a Data Word
T	PW 0 ..255			Transfer the contents of ACCU 1 to a Peripheral Word
T	ID 0 .. 124	0,7	0,4	Transfer the contents of ACCU 1 to an Input Double Word
T	QD 0..124	0,7	0,4	Transfer the contents of ACCU 1 to an Output Double Word
T	FD 0..252	0,7	0,4	Transfer the contents of ACCU 1 to a Flag Double Word
T	SD 0..4092	0,9	0,5	Transfer the contents of ACCU 1 to a Special Double Flag Word
T	DD 0..252	1,8	0,9	Transfer the contents of ACCU 1 to a Data Word
TIR	0 (ACCU1) 2 (ACCU2)	0,8	0,3	Transfer the register contents (addressed by ACCU 1) indirectly into the memory word
TDI	A1 (ACCU1) A2 (ACCU2)	1,0	0,4	Transfer the register contents (addressed by ACCU 1) indirectly into the memory word
TNB	RS 0..255	0,45 /Byte	0,15 /Byte	Transfer a Block Byte (Number 0..255) End address source : ACCU2 (16 Bit) End address target : ACCU1 (16 Bit)
T	RS 0..255	0,7	0,3	Transfer a System Data Word
T=	<Parameter>	1,5	0,7	Transfer the contents of ACCU 1 to the formal Operand (parameter type: BY,W,D)

4.6.5 Timer operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
SP	T 0..255	1,0	0,5	Start a Timer (stored in ACCU 1) as Signal contracting pulse on the leading edge of the RLO
SE	T 0 ..255	1,0	0,5	Start a Timer (stored in ACCU 1) as extended pulse (signal stretching) on the leading edge of the RLO
SR	T 0..255	1,0	0,5	Start an on-delay Timer (stored in ACCU 1) on the leading edge of the RLO
SS	T 0 ..255	1,0	0,5	Start a stored on-delay Timer (stored in ACCU 1) on the leading edge of the RLO
SF	T 0..255	1,0	0,5	Start an off-delay Timer (stored in ACCU 1) on the trailing edge of the RLO
R	T 0 ..255	1,0	0,5	Reset a Timer if RLO= "1"
FR	T 0..255	1,0	0,5	Enable a Timer for cold restart. If RLO="1" "FR T" restarts the Timer
SP=	<Parameter>	2,0	0,9	Start a Timer (formal operand) as a pulse with the value stored in ACCU 1 (FB - Parameter: T)
SEC=	< Parameter >	2,0	0,9	Start a Timer (formal operand) as an extended pulse with the value stored in ACCU 1 (FB - Parameter: T)
SD=	< Parameter >	2,0	0,9	Start an on-delay Timer (formal operand) with the value stored in ACCU 1(FB - Parameter: T)
SSU=	< Parameter >	2,0	0,9	Start a stored on-delay Timer (formal operand) with the value stored in ACCU 1 (FB - Parameter: T)
SFD=	< Parameter >	2,0	0,9	Start an off-delay Timer (formal operand) on the trailing edge of the RLO with the value stored in ACCU1 (FB - Parameter: T)
FR=	< Parameter >	2,0	0,9	Enable formal operand (Timer) for cold restart (FB - Parameter: T)
RD=	< Parameter >	2,0	0,9	Reset the formal operand for a Timer (FB parameter: T)

4.6.6 Counter Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
CU	C 0..255	0,9	0,5	Counter counts up 1 on the leading edge of the RLO
CD	C 0..255	0,9	0,5	Counter counts down 1 on the leading edge of the RLO
S	C 0..255	0,9	0,5	Set counter if RLO = "1"
R	C 0..255	0,9	0,5	Reset counter if RLO = "1"
FR	C 0..255	0,9	0,5	Enable a Counter for cold restart. If RLO="1" "FR C" sets, decrements, or increments the Counter.

4.6.5 Counter operations (continued)

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
FR=	< Parameter >	2,0	0,9	Enable formal operand (Counter) for cold restart (FB - Parameter: C)
RD=	< Parameter >	2,0	0,9	Reset the formal operand for a Counter (FB parameter: C)

4.6.7 Arithmetic Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
+F	-	0,6	0,25	ACCU1 = ACCU2 + ACCU1 (16 Bit)
-F	-	0,6	0,25	ACCU1 = ACCU2 - ACCU1 (16 Bit)
xF	-	1,0	0,6	ACCU1 = ACCU2 * ACCU1 (16 Bit)
:F	-	1,5	1,0	ACCU1 = ACCU2 / ACCU1 (16 Bit)
+D	-	0,8	0,35	ACCU1 = ACCU2 + ACCU1 (32 Bit)
-D	-	0,8	0,35	ACCU1 = ACCU2 - ACCU1 (32 Bit)
D	0 .. 255	0,4	0,2	ACCU1 = ACCU2 - Value (8 Bit)
I	0 .. 255	0,4	0,2	ACCU1 = ACCU2 + Value(8 Bit)
ADD	BN -128 .. +127	0,4	0,2	ACCU1 = ACCU2 + BN Value (8 Bit) (Extended Sign)
ADD	KF - 32768..+32767	0,4	0,2	ACCU1 = ACCU2 +/- Value (16 Bit)
ADD	DH 0.. FFFFFFF	0,9	0,4	ACCU1 = ACCU2 + Value (32 Bit)

4.6.8 Comparison Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
!=F	-	0,7	0,3	RLO = ACCU2 = ACCU1 (16 Bit)
><F	-	0,7	0,3	RLO = ACCU2 >< ACCU1 (16 Bit)
>F	-	0,7	0,3	RLO = ACCU2 > ACCU1 (16 Bit)
>=F	-	0,7	0,3	RLO = ACCU2 >= ACCU1 (16 Bit)
<F	-	0,7	0,3	RLO = ACCU2 < ACCU1 (16 Bit)
<=F	-	0,7	0,3	RLO = ACCU2 <= ACCU1 (16 Bit)
!=D	-	0,7	0,3	RLO = ACCU2 = ACCU1 (32 Bit)
><D	-	0,7	0,3	RLO = ACCU2 >< ACCU1 (32 Bit)
>D	-	0,7	0,3	RLO = ACCU2 > ACCU1 (32 Bit)
>=D	-	0,7	0,3	RLO = ACCU2 >= ACCU1 (32 Bit)
<D	-	0,7	0,3	RLO = ACCU2 < ACCU1 (32 Bit)
<=D	-	0,7	0,3	RLO = ACCU2 <= ACCU1 (32 Bit)

4.6.9 Block Call Operations

4.6.9.1 Absolute Calls

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
SPA	PB 0 .. 255	2,3	1,3	Jump unconditionally to a Program Block
SPA	FB 0 ..255	2,3	1,3	Jump unconditionally to a Function Block
BA	FX 0 .. 255	2,3	1,3	Jump unconditionally to a Extended Function Block
SPA	SB 0 ..255	2,3	1,3	Jump unconditionally to a Sequence Block
SPA	OB 1 .. 255	2,3	1,3	Call an Organization Block unconditionally
DO=	<Parameter>			Call a Block specified as actual Operands (Process Block Parameter)

4.6.9.2 Conditional Calls (if RLO=1)

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
SPB	PB 0 .. 255	2,6	1,4	Jump conditionally to a Program Block
SPB	FB 0 ..255	2,6	1,4	Jump conditionally to a Function Block
BAB	FX 0 .. 255	2,6	1,4	Jump conditionally to a Extended Function Block
SPB	SB 0 ..255	2,6	1,4	Jump conditionally to a Sequence Block
SPB	OB 1 .. 255	2,6	1,4	Call an Organization Block conditionally

4.6.9.3 Data Block Calls

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
A	DB 1 .. 255			Call a Data Block
AX	DX 0 ..255			Call an Extended Data Block
E	DB 1 .. 255			Generate a Data Block. (No. of Data Block deposited in ACCU 1)
EX	DX 0 ..255			Generate an Extended Data Block. (No. of Data Block deposited in ACCU 1)

4.6.10 Block Return Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
BE	-	2,2	0,9	Block end (termination of a Block)
BEB	-	3,0	1,5	Block end, conditional RLO = 1
BEA	-	2,2	0,9	Block end, unconditional

4.6.11 No Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
NOP	0 .. 1	0,35	0,15	No operation (all bits set / reset)
BLD	0 .. 255	0,5	0,2	Display generation operation

4.6.12 Stop Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
STP	-			Stop: Scanning cycle is not completed. Error ID "STS" is set in the I-Stack.
STS	-			Stop operation. Program processing is interrupted immediately after this operation.

4.6.13 Bit Operations

4.6.13.1 Test a Bit for "1"

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
TB	I 0.0 .. 127.7	1,7	0,9	Test an Input
TB	Q 0.0..127.7	1,7	0,9	Test an Output
TB	F 0.0..255.7	1,7	0,9	Test a Flag
TB	T 0.0..255.15	1,7	0,9	Test a Timer Word Bit
TB	C 0.0..255.15	1,7	0,9	Test a Counter Word Bit
TB	D 0.0..255.15	1,7	0,9	Test a Data Word Bit
TB	RS 0.0..255.15	1,7	0,9	Test a Data Word Bit in the System data range

4.6.13.2 Test a Bit for "0"

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
PN	I 0.0 .. 127.7	1,7	0,9	Test an Input
PN	Q 0.0..127.7	1,7	0,9	Test an Output
PN	F 0.0..255.7	1,7	0,9	Test a Flag
PN	T 0.0..255.15	1,7	0,9	Test a Timer Word Bit
PN	C 0.0..255.15	1,7	0,9	Test a Counter Word Bit
PN	D 0.0..255.15	1,7	0,9	Test a Data Word Bit
PN	RS 0.0..255.15	1,7	0,9	Test a Data Word Bit in the System data range

4.6.13.3 Set a Bit

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
SU	I 0.0 .. 127.7	1,7	0,9	Set an Input
SU	Q 0.0..127.7	1,7	0,9	Set an Output
SU	F 0.0..255.7	1,7	0,9	Set a Flag
SU	T 0.0..255.15	1,7	0,9	Set a Timer Word Bit
SU	C 0.0..255.15	1,7	0,9	Set a Counter Word Bit
SU	D 0.0..255.15	1,7	0,9	Set a Data Word Bit
SU	RS 0.0..255.15	1,7	0,9	Set a Data Word Bit in the System data range

4.6.13.4 Reset a Bit

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
RU	I 0.0 .. 127.7	1,7	0,9	Reset an Input
RU	Q 0.0..127.7	1,7	0,9	Reset an Output
RU	F 0.0..255.7	1,7	0,9	Reset a Flag
RU	C 0.0..255.15	1,7	0,9	Reset a Timer Word Bit
RU	Z 0.0..255.15	1,7	0,9	Reset a Counter Word Bit
RU	D 0.0..255.15	1,7	0,9	Reset a Data Word Bit
RU	RS 0.0..255.15	1,7	0,9	Reset a Data Word Bit in the System data range

4.6.14 Conversion Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
CFW	-	0,4	0,15	Form the one's complement of ACCU 1 (16 Bit)
CSW	-	0,6	0,3	Form the two's complement of ACCU 1 (16 Bit) CC 1 / CC 0 and OV are affected
CSD	-	0,8	0,4	Form the two's complement of ACCU 1 (32 Bit)
DEF	-	1,7	0,8	Convert a fixed point value into BCD format (16 Bit)
DUF	-	2,3	1,3	Convert a BCD format into a fixed point value (16 Bit)
DED	-	3,5	1,2	Convert a fixed point value into BCD format (32 Bit)
DUD	-	5,5	3,5	Convert a BCD format into a fixed point value (32 Bit)

4.6.15 Shift and Rotation Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
SLW	n = 0..15	0,9	0,4	Shift the contents of ACCU 1 to the left by the value n (16 Bit)
SRW	n = 0..15	0,9	0,4	Shift the contents of ACCU 1 to the right by the value n (16 Bit)
SLD	n = 0..32	0,9	0,4	Shift the contents of ACCU (double word) to the left by the value n (32 Bit)
SSW	n = 0..15	0,9	0,4	Shift the contents of ACCU 1 with the sign to the left by the value n (16 Bit)
SSD	n = 0..32	0,9	0,4	Shift the contents of ACCU 1 with the sign to the right by the value n (16 Bit)
RLD	n = 0..32	0,9	0,4	Rotate the contents of ACCU (double word) to the left t by the value n (32 Bit)
RRD	n = 0..32	0,9	0,4	Rotate the contents of ACCU (double word) to the right by the value n (32 Bit)

4.6.16 Jump Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
JU=	<Label> max. 4 characters	0,7	0,3	Jump unconditionally to the label.
JC=	<Label> max. 4 characters	0,9	0,5	Jump conditionally (RLO = 1) to the label. (If the RLO is "0", it is set to "1").
JZ=	<Label> max. 4 characters	0,9	0,5	Jump if the result (CC 1 = 0 and CC 0 = 0) is zero. The RLO is not changed.
JN=	<Label> max. 4 characters	0,9	0,5	Jump if the result (CC 1 <> CC 0) is not zero. The RLO is not changed
JP=	<Label> max. 4 characters	0,9	0,5	Jump if the result (CC 1 = 1 and CC 0 = 0) is > 0. The RLO is not changed.
JM=	<Label> max. 4 characters	0,9	0,5	Jump if the result (CC 1 = 0 and CC 0 = 1) is < 0. The RLO is not changed.
JO=	<Label> max. 4 characters	0,9	0,5	Jump on overflow (OVERFLOW bit set). The RLO is not changed.
JOS=	<Label> max. 4 characters	0,9	0,5	Jump on latching overflow (OVERFLOW bit set). The RLO is not changed.
JUR	-32768..+32767	0,9	0,5	Relative Jump within a Function Block

4.6.17 Other Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
IA	-	0,9	0,4	Disable Interrupt
RA	-	6,0	3,0	Enable Interrupt
DO	DW 0 ..255			Process Data Word. The next operation is combined through logic OR with the parameter specified in the Data Word and executed.

4.6.17 Other operations (continued)

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
DO	FW 0 ..255			Process Flag Word. The next operation is combined through logic OR with the parameter specified in the Data Word and executed.
DI	-			Process via a formal operand (indirectly). The number of the formal operand is in ACCU 1
TAK	-	0,6	0,3	Swap the contents of ACCU 1 and ACCU 2
DO	RS 10 ..			*SoftPLC* Special Instructions

4.7 Memory Allocation of the *SoftPLC*43

The Memory of the *SoftPLC*43 emulates all-important features of the 115U CPU 943 Memory.

Attention:

If accessing the *SoftPLC*43 Memory from a Windows application the Low Byte and the High Byte must be swapped.

Memory Allocation PLC43:

Address	Length Kbytes	Function
0000	4096	S Flags
1000	49152	Block Buffer
D000	2048	reserved
D800	1025	Internal Blocks
DC00	512	OB Address List
DE00	512	FB Address List
E000	512	PB Address List
E200	512	SB Address List
E400	512	DB Address List
E600	512	FX Address List
E800	512	DX Address List
EA00	512	System Data
EC00	256	Timers 0..127
ED00	256	Counters 0..127
EE00	256	Flags
EF00	128	Process Input Image
EF80	128	Process Output Image
F000	256	Timers 128..255
F100	256	Counters 128..255
F200	512	Transfer area for special procedures B RS 28
F400	512	Miscellaneous Values for Timers (reserved)
F600	2560	reserved

4.8 Address Allocation in the System Data Area

The System Data Area of the *SoftPLC*43 emulates all the important features of the 115U CPU 943 System Data Area.

Unlisted System Data Addresses are not used and reserved for future use.

System Data area structure *SoftPLC*43:

Address (hex.)	System Data Word	Bit	Description
EA08	4	8	Stop at the end of a program execution
EA0A	5	10 11 12 13	Compression aborted Address List Structure Block Shift active Block Shift requested
EA0C	6	2 13 14 15	Alarm enable PLC in Start condition Stop Display Stop Condition from Programming Device *(S5 for Windows*)
EA0E	7	5 6 13	Start not possible Synchronization Failure (Blocks are corrupted) Block Header corrupted
EA10.. EA16	8.. 11		Integrated Clock
EAC0	96		Scan monitoring time (multiple of 10ms)
EAC2	97		Interval timer for OB13 (multiple of 10ms)
EAC4	98		Interval timer for OB12 (multiple of 10ms)
EAC6	99		Interval timer for OB11 (multiple of 10ms)
EAC8	100		Interval timer for OB10 (multiple of 10ms)
EACA	101		Time (in ms) calling OB6
EAF2	121		Actual scan time (in ms)
EAF4	122		Maximum scan time (in ms)
EAF6	123		Minimum scan time (in ms)
EBF0.. EBFF	248.. 255		Reserved for user

The following System Data is only valid if the *SoftPLC*43 is in a Stop condition.

Address (hex.)	System Data Word	Bit	Description
EB00	128		ACCU 1 (High Word)
EB02	129		ACCU 2 (High Word)
EB96	203		ACCU 1 (Low Word)
EB98	204		ACCU 2 (Low Word)
EB9A	205		Statement Register
EB9C	206		Step Address Counter
EB9E	207		Block Stack Pointer
EBA0	208		Start address of the Data Block
EBA2.. EBA8	209.. 212		Bracket Stack (nesting level)
EBAA	213		Interrupt Stack Result Display
		0	First Scan
		1	RLO
		3	OR Memory
		4	Latched Overflow
		5	Overflow
		6	Anz 0 (ACCU 1 Shift)
		7	Anz 1 (ACCU 1 Shift)
EBAC	214		Interrupt Stack Cause of Fault
		2	I/O's not ready
		3	No Warm Start possible. Cold Start requested
		4	Scan time exceeded
		8	Error in the CPU self test routine
		9	Block Stack overflow
		10	Operation interrupted by a programmed Stop
		11	Statement cannot be interpreted
		12	Transfer error for Data Block Statements
		13	Substitution failure

4.9 Organization Blocks integrated in the PLC43

OB 19	Response when calling a not loaded Block
OB 27	Response to substitution errors
OB 31	Scan Time triggering
OB 32	Response to transfer / load errors
OB 250	Operating system service routine
OB 251	PID control algorithm

4.9.1 Response when Calling a not Loaded Block (OB 19)

In OB 19 you can program the response of the *SoftPLC* when a not loaded Block is called. If OB19 is not programmed the *SoftPLC* continues the execution (no response) directly after the jump statement. OB19 may be used to put the *SoftPLC* in the STOP mode when calling a not loaded Block.

4.9.2 Response to Substitution Errors (OB 27)

A substitution error (SUF) can occur when a formal operand has been modified after the PLC program Block was called (JU FBx). When a substitution error occurs, the *SoftPLC* normally goes into the STOP mode. If OB 27 is present the *SoftPLC* is executing the OB 27 instead of going into the STOP mode.

4.9.3 Scan Time triggering (OP 31)

A scan time monitor is integrated. If the cycle time of a program exceeds a preset time (e.g. 600msec), the CPU is forced into the “Stop” mode.

The maximum allowable cycle time can be set in the system data word 96 or in the DB1.

4.9.4 Response to Transfer / Load Errors (OB 32)

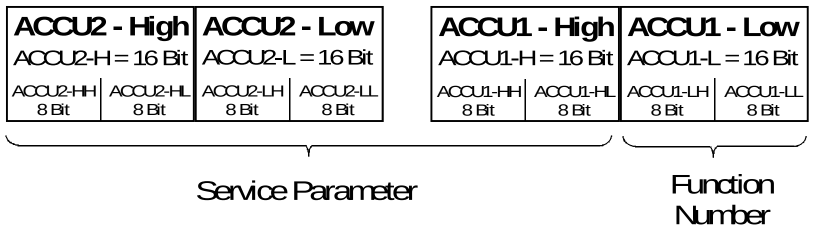
When a transfer / load error (TRAF occurs, the *SoftPLC* normally goes into the STOP mode. If OB 32 is present the *SoftPLC* is executing the OB 32 instead of going into the STOP mode.

4.9.5 Operating System Service Routine (OB 250)

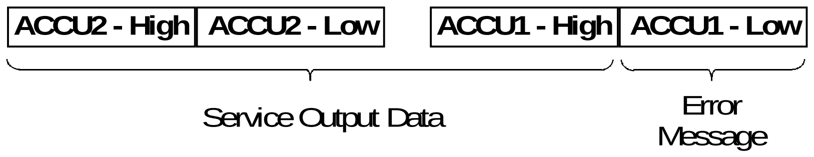
With the operating system service routine, you have the ability to activate various operating system functions or you can modify certain parameters during the cyclic PLC program execution.

A function number is assigned to each operating system service routine. This function number must be loaded into ACCU1 (Low Byte).

The service parameters are loaded into ACCU2 and /or ACCU1 (High Byte)



Error messages are posted to ACCU1 (Low Byte) with the service routine parameters posted to ACCU2 and /or ACCU1 (High Byte).



The following functions are integrated in the *SoftPLC*43:

Operating System Service Routine	Function Number	Parameter	Error Message
Activating OB 6	1	ACCU2-L: 0: Cancel activation 1 ... 65535: Time value (msec)	ACCU1-L: 0: no Error
New Interval for OB 10	2	ACCU2-L: 0: no OB 10 call 1 ... 65535: Time value (msec)	ACCU1-L: 0: no Error
New Interval for OB 11	3	ACCU2-L: 0: no OB 11 call 1 ... 65535 Time value (msec)	ACCU1-L: 0: no Error
New Interval for OB 12	4	ACCU2-L: 0: no OB 12 call 1 ... 65535 Time value (msec)	ACCU1-L: 0: no Error
New Interval for OB 13	5	ACCU2-L: 0: no OB 13 call 1 ... 65535 Time value (msec)	ACCU1-L: 0: no Error

The following functions are integrated in the *SoftPLC*43: (continued)

Operating System Service Routine	**Function Number**	**Parameter**	**Error Message**
Generating a DB without TRAF	10	ACCU2-LL: 0 ... 255: DB Number ACCU2-H: 0: delete DB 1 ... FFF9: DB size until including DW	ACCU1-L: 0: no Error 5: TRF (identical to EDB)
Generating a DX without TRAF	11	ACCU2-LL: 0 ... 255: DX Number ACCU2-H: 0: delete DX 1 ... FFF9: DX size until including DW	ACCU1-L: 0: no Error 5: TRF (identical to EXDX)
Address (Byte) reading from the S5 Bus	13	ACCU2-L: 0 ... FFFF: Address	ACCU1-L: 0: no Error 5: QVZ ACCU2-LL Byte read
Address (Byte) writing to the S5 Bus	14	ACCU2-L: 0 ... FFFF: Address ACCU2-HL: 0 ... FF: Byte to be written	ACCU1-L: 0: no Error 5: QVZ
Address (Word) reading from the S5 Bus	15	ACCU2-L: 0 ... FFFF: Address	ACCU1-L: 0: no Error 5: QVZ 7: invalid Address ACCU2-L Word read
Address (Word) writing to the S5 Bus	16	ACCU2-L: 0 ... FFFF: Address ACCU2-HL: 0 ... FF: Word to be written	ACCU1-L: 0: no Error 5: QVZ 7: invalid Address

4.9.6 PID control algorithm (OB 251)

The *SoftPLC* has an integrated PID control algorithm (OB 251). The execution time in the *SoftPLC* is about 100 times faster than the execution time in a hardware PLC. To enhance the accuracy of the PID control algorithm the *SoftPLC* uses ***REAL*** values for the calculations. Prior to calling the OB251 the PID control Data Block (DB) holding the controller parameters and other controller specific data must be open. The PID control algorithm is called periodically (sampling interval) and calculates the new variables.

5 *SoftPLC* PLC 45

In the following chapter the differences between the *SoftPLC*45 and the Siemens CPU945 are described. Also, all the instructions the *SoftPLC*45 can execute (operations list, special Instructions) and the memory allocation are listed.

5.1 Additional Features in the *SoftPLC*45

The following additional features are integrated in the *SoftPLC*45.

- Additional Special Instructions DO RS 10 .. DO RS 27
- The PID Block OB251 is implemented. The execution is extremely fast. About 100 µs with a 486-66 PC.
- Customer selectable use of four (4) accumulators. Register instructions compatible to the Siemens CPU 948

5.2 Differences between the *SoftPLC*45 and the Siemens CPU 945

- ***"Clear All Blocks"*** from the PLC Block List executes an **Overall Reset.**
- An endless loop within a **Disable Interrupt** will hang up the PC program.
- If an Enable Interrupt is not programmed, it is automatically inserted at the end of the OB.

5.3 Functions not Incorporated in the *SoftPLC*45

Due to the physical differences between a hardware PLC and a PC executing a Windows application, not all of the hardware functions can be realized in the *SoftPLC*45 software, however a program written to be executed an a PLC U115 should run on the *SoftPLC* with a minimum of modifications.

The following functions are not realized with the *SoftPLC*:

- Flags are non retentive (also see chapter 3.1.7)
- Only the Following System Data Words have their integrated functions.

- SD 8..12 (Integral Clock)

- SD 96	(Scan Monitoring Time)
- SD 97..100	(Time controlled program scanning for OB10..OB13)
- SD 101	(Interrupt generated by internal timers for OB6)
- SD 121..123	(Scan Time)
- SD 203..238	(Interrupt Stack)

All other System Data Words are not used in the *SoftPLC*.

- Timer OB's and Interrupt OB's are not interrupted by Windows. Therefore these OB's may not have a long execution time (<1ms).
- The OB6 can not interrupt the Timer.
- The Data Handling Blocks FB244..FB249 are not available.
- The integrated Clock has no alarm function. The operating hours counter and the input of a correction value to compensate for clock inaccuracy are not implemented. The System Data Area of the *SoftPLC* is equal to the System Data Area of the CPU 945. The above mentioned functions have no meaning in the *SoftPLC*.
- No setting of parameters for internal functions via DB1.

5.4 Operations of the PLC45

The execution times specified are guideline values for the instructions. The actual execution time of an instruction depends on the processor used and its speed. To calculate the execution time of a PLC Program, the time slice setting of the *SoftPLC* must be taken into consideration.

In a **single processor system** the time slice is set to 50%, the PC is executing the *SoftPLC* 50% of its time and the other 50% is used by Windows. You have to multiply the calculated execution time of a PLC program by two.

In a **dual processor system,** one CPU is completely allocated to the *SoftPLC*. No time slice has to be taken into consideration.

Having short PLC programs, you must consider that the *SoftPLC* is called with a fixed time pulse (2 ms). The time pulse will start the execution of OB1. With one time pulse call, the OB1 will be executed only once.

5.5 *SoftPLC*45 Special Instructions

The following special instructions are implemented in the *SoftPLC*45 (Windows NT version):

- DO RS 12 : Assigning Peripheral Bytes to a Hardware Ports
- DO RS 17 : Read from a Hardware Port (single bytes direct)

- DO RS 30 : Read from a Hardware Port (single words direct)
- DO RS 18 : Write into a Hardware Port (single bytes direct)
- DO RS 31 : Write into a Hardware Port (single words direct)
- DO RS 20 : Read from a Memory Location and store the Contents in the Low Byte of ACCU 1
- DO RS 21 : Write the Low Byte from ACCU2 into a Memory Location
- DO RS 22 : Load Data into multiple Memory Locations of the 115U Memory
- DO RS 23 : Store Data from multiple Memory Locations of the 115U Memory
- DO RS 28 : Call a C Program procedure

5.5.1 DO RS 12 : Assigning Peripheral Bytes to a Hardware Ports

Call:

A DB <AssignmentDB>

L KF <DW Address of the 1st Assignment (n)>

L KF <Number of Assignments>

DO RS 12

AssignmentDB Structure:

DRn =Peripheral Byte No. (0..255)

DLn = 0: read + write

1: read only

2: write only

DWn+1 = Hardware Port address

DWn+2 = next Assignment

5.5.2 DO RS 17 : Read from a Hardware Port (single bytes direct)

Call:

L KH <Hardware Port Address>

DO RS 17

After the Call:

ACCU1: Data Byte read

5.5.3 DO RS 18 : Write into a Hardware Port (single bytes direct)

Call:

L <Data Byte to be written to the Port>

L KH <Hardware Port Address>

DO RS 18

5.5.4 DO RS 20 : Read a Memory Location and store the Contents in the Low Byte of ACCU 1

Call:

L DH <Source address>(Linear 32 Bit address)

Example: 000D0000 → D000:0

DO RS 20

After the Call:

ACCU1 (low Byte) : Data Byte read

5.5.5 DO RS 21 : Write the Low Byte from ACCU2 into a Memory Location

Call:

L <Data Word>

L DH <Destination address>(Linear 32-Bit address)

Example: 000D0000 → D000:0

DO RS 21

5.5.6 DO RS 22 : Load Data into multiple Memory Locations of the 115U Memory

Call:

L DH <Source address>(Linear 32-Bit Address)

Example: 000D0000 → D000:0

L DH <Length, Address at the 115U Memory>

DO RS 22

The address for the 115U is built from the lower 5 hexadecimal digits. The upper 3 hexadecimal digits define the length.

The *SoftPLC*45 emulates the Memory of the PLC 115U (CPU 945). For more details see chapter 5.7 of this manual.

Example:

```
DH 000D0000
DH FA011000
DO RS 22
```

Transfer 4000 Bytes starting from address D000:0000 to the 115U Memory starting at the memory location 11000h.

5.5.7 DO RS 23 : Store Data from multiple Memory Locations of the 115U Memory

Call:

L DH	<Length, Address at the 115U Memory>
L DH	<Destination address>(Linear 32-Bit Address)
	Example: 000D0000 → D000:0
DO RS 23	

5.5.8 DO RS 28 : Call a C program procedure

Call:

L KH	<C procedure number>
DO RS 28	

The *SoftPLC* software can handle up to 256 C procedures. In the directory **\CB123** you will find an example of a C procedure call. To compile the C program the Microsoft Visual C++ version 4.x is required.

The C programs are linked to the *SoftPLC* software whenever the *SoftPLC* software starts to be executed. With the start of the *SoftPLC* software the *SoftPLC* directory is searched for the DLL’s **CB000.DLL** up to **CB255.DLL**. The DLL’s recognized are linked into the *SoftPLC* software. If you compiled a C program again, you must restart the *SoftPLC* software to link the new compiled C program into the *SoftPLC* software.

Each C procedure exports a C function:

```
Extern “C” void C_PROGRAMM (TCPU * cpu)
```

With the parameter “cpu” from the C function you have access to the ACCU1, ACCU2, and the 115 memory structure.

Attention:

The C procedure is executed with the highest Windows priority. The execution time of the C procedure directly influences the execution time of the *SoftPLC* software and therefore takes influences of the real time capability of the *SoftPLC*. Please consider the following rules:

- Do not call any Windows function
- Do not call a function requiring operators intervention
- Call only functions having a defined execution time

5.5.9 DO RS 30 : Read from a Hardware Port (single words direct)

Call:

L KH <Hardware Port Address>

DO RS 30

After the Call:

ACCU1: Data word read

5.5.10 DO RS 31 : Write into a Hardware Port (single words direct)

Call:

L <Data Byte to be written to the Port>

L KH <Hardware Port Address>

DO RS 18

5.6 *SoftPLC* PLC45 Standard Instructions

The following standard instructions are implemented in the *SoftPLC*45 (Windows NT version):

5.6.1 Binary Scan Operations

5.6.1.1 Scan Operand for “1” and combine with RLO through logic AND

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
A	I 0.0 .. 127.7	0,9	0,4	Scan Input
A	Q 0.0..127.7	0,9	0,4	Scan Output
A	F 0.0..255.7	0,9	0,4	Scan Flag
A	S 0.0..4095.7	1,4	0,6	Scan S Flag
A	D 0.0..255.15	1,6	0,8	Scan Data Word
A	T 0..255	1,3	0,6	Scan Timer
A	C 0..255	1,1	0,6	Scan Counter
A=	<Parameter>	1,9	0,9	Scan FB Parameter (Type : BI)

5.6.1.2 Scan Operand for “0” and combine with RLO through logic AND

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
UN	I 0.0 .. 127.7	0,9	0,4	Scan Input
UN	Q 0.0..127.7	0,9	0,4	Scan Output
UN	F 0.0..255.7	0,9	0,4	Scan Flag
UN	S 0.0..4095.7	1,4	0,6	Scan S Flag
UN	D 0.0..255.15	1,6	0,8	Scan Data Word
UN	T 0..255	1,3	0,6	Scan Timer
UN	C 0..255	1,1	0,6	Scan Counter
UN=	<Parameter>	1,9	0,9	Scan FB Parameter (Type : BI)

5.6.1.3 Scan Operand for "1" and combine with RLO through logic OR

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
O	I 0.0 .. 127.7	0,9	0,4	Scan Input
O	Q 0.0..127.7	0,9	0,4	Scan Output
O	F 0.0..255.7	0,9	0,4	Scan Flag
O	S 0.0..4095.7	1,4	0,6	Scan S Flag
O	D 0.0..255.15	1,6	0,8	Scan Data Word
O	T 0..255	1,3	0,6	Scan Timer
O	C 0..255	1,1	0,6	Scan Counter
O=	<Parameter>	1,9	0,9	Scan FB Parameter (Type : BI)

5.6.1.4 Scan Operand for "0" and combine with RLO through logic OR

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
ON	I 0.0 .. 127.7	0,9	0,4	Scan Input

ON	Q 0.0..127.7	0,9	0,4	Scan Output

5.6.1.4 Scan Operand for “0” and combine with RLO through logic OR (continued)

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
ON	F 0.0..255.7	0,9	0,4	Scan Flag
ON	S 0.0..4095.7	1,4	0,6	Scan S Flag
ON	D 0.0..255.15	1,6	0,8	Scan Data Word
ON	T 0..255	1,3	0,6	Scan Timer
ON	C 0..255	1,1	0,6	Scan Counter
ON=	<Parameter>	1,9	0,9	Scan FB Parameter (Type : BI)

5.6.1.5 Other Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
O	-	2,3	1,1	Combine AND operations through logic OR
A(	-	2,3	1,1	Combine expressions enclosed in parentheses through logic AND (8 levels possible)
O(	-	2,3	1,1	Combine expressions enclosed in parentheses through logic OR (8 levels possible)
)	-	2,3	1,1	Closing parenthesis
AW	-	0,5	0,2	ACCU1=ACCU1 AND ACCU2 (16 Bit)
OW	-	0,5	0,2	ACCU1=ACCU1 OR ACCU2 (16 Bit)
XOW	-	0,5	0,2	ACCU1=ACCU1 XOR ACCU2 (16 Bit)

5.6.2 SET / RESET Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
S	I 0.0 .. 127.7	1,0	0,6	Set Input
S	Q 0.0..127.7	1,0	0,6	Set Output
S	M 0.0..255.7	1,0	0,6	Set Flag
S	S 0.0..4095.7	1,4	0,8	Set S Flag
S	D 0.0..255.15	1,9	0,9	Set Data Word
S=	<Parameter>	1,9	0,9	Set FB Parameter (Type : BI)
R	I 0.0 .. 127.7	1,0	0,6	Reset Input
R	Q 0.0..127.7	1,0	0,6	Reset Output
R	M 0.0..255.7	1,0	0,6	Reset Flag
R	S 0.0..4095.7	1,4	0,8	Reset S Flag
R	D 0.0..255.15	1,9	0,9	Reset Data Word
RB=	<Parameter>	1,9	0,9	Reset FB Parameter (Type : BI)
=	I 0.0 .. 127.7	1,0	0,5	Assign Input
=	Q 0.0..127.7	1,0	0,5	Assign Output
=	M 0.0..255.7	1,0	0,5	Assign Flag
=	S 0.0..4095.7	1,4	0,7	Assign S Flag
=	D 0.0..255.15	1,9	0,9	Assign Data Word
==	<Parameter>	1,9	0,9	Assign FB Parameter (Type : BI)

5.6.3 Load Operations

5.6.3.1 Load Variable

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
L	IB 0 .. 127	0,5	0,2	Load Input Byte
L	QB 0..127	0,5	0,2	Load Output Byte
L	FB 0..255	0,5	0,2	Load Flag Byte
L	SY 0..4095	0,7	0,3	Load S Flag Byte
L	DL 0..255	1,5	0,8	Load Data Byte (left Byte)
L	DR 0..255	1,5	0,8	Load Data Byte (right Byte)
L	PY 0 ..255			Load Peripheral Byte
L	IW 0 .. 126	0,6	0,25	Load Input Word
L	QW 0..126	0,6	0,25	Load Output Word
L	FW 0..254	0,6	0,25	Load Flag Word
L	SW 0..4094	0,8	0,3	Load S Flag Word
L	DW 0..254	1,6	0,8	Load Data Word
L	PW 0 ..255			Load Peripheral Word
L	ID 0 .. 124	0,7	0,3	Load Input Double Word
L	QD 0..124	0,7	0,3	Load Output Double Word
L	FD 0..252	0,7	0,3	Load Flag Double Word
L	SD 0..4092	0,9	0,4	Load S Flag Double Word
L	DD 0..252	1,8	0,9	Load Data Double Word
L	T 0 ..255	0,7	0,3	Load Time (Binary Code)
L	C 0..255	0,7	0,3	Load Count (Binary Code)
LD	T 0 ..255	2,5	1,5	Load Time (BCD Code)
LD	C 0..255	2,5	1,5	Load Count (BCD Code)
L	RS 0 ..255	0,5	0,2	Load a System Data Word
LIR	0 (ACCU1) 2 (ACCU2)	0,7	0,3	Load the contents of a Memory Word (addressed by ACCU1) indirectly into the Register (0: ACCU 1; 2: ACCU 2)
LDI	A1 (ACCU1) A2 (ACCU2)	0,7	0,3	Load the contents of a Memory Word (addressed by 32 Bit ACCU1) indirectly into the Register (A1 = ACCU 1; A2 = ACCU 2)
L=	<Parameter>	1,5	0,65	Load the value of the formal operand into ACCU 1 (parameter type: BY, W, DW)
LC=	<Parameter>	4,0	2,0	Load the value of the formal operand in BCD code into ACCU 1 (parameter type: T,Z)
LW=	<Parameter>	0,8	0,35	Load a formal operand bit pattern into ACCU 1 (parameter type : D; Data type: KF,KH,KM,KY,KC,KT,KZ)

5.6.3.2 Load constant

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
L	KB 0 .. 255	0,4	0,2	Load a constant into ACCU 1 (Bit 0..7) and 0 into Bit 8..15
L	KS ‘AA’	0,4	0,2	Load a constant (2 characters in ASCII format) into ACCU 1

5.6.3.2 Load constant (continued)

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
L	KF -32768 .. +32767	0,4	0,2	Load a constant (fixed point number) into ACCU1
L	KH 0..FFFF	0,4	0,2	Load a constant (hexadecimal) into ACCU 1
L	DH 0 .. FFFFFFFF	0,7	0,3	Load a constant (hexadecimal double word)
L	KM <16 Bit pattern>	0,4	0,2	Load a constant (bit pattern) into ACCU 1
L	KT 0.0 .. 999.3	0,4	0,2	Load a constant (Timer value) into ACCU 1 (BCD coded)
L	KY 0 .. 255, 0..255	0,4	0,2	Load a constant (2 byte number) into ACCU 1
L	KC 0 .. 999	0,4	0,2	Load a constant (Counter value) into ACCU 1 (BCD coded)

5.6.4 Transfer Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
T	IB 0 .. 127	0,5	0,2	Transfer the contents of ACCU 1 to an Input Byte
T	QB 0..127	0,5	0,2	Transfer the contents of ACCU 1 to an Output Byte
T	FB 0..255	0,5	0,2	Transfer the contents of ACCU 1 to a Flag Byte
T	SY 0..4095	0,7	0,3	Transfer the contents of ACCU 1 to a Special Flag Byte
T	DL 0..255	1,5	0,8	Transfer the contents of ACCU 1 to a Data Word (left Byte)
T	DR 0..255	1,5	0,8	Transfer the contents of ACCU 1 to a Data Word (right Byte)
T	PY 0 ..255			Transfer the contents of ACCU 1 to a Peripheral Byte
T	IW 0 .. 126	0,5	0,3	Transfer the contents of ACCU 1 to an Input Word
T	QW 0..126	0,5	0,3	Transfer the contents of ACCU 1 to an Output Word
T	FW 0..254	0,5	0,3	Transfer the contents of ACCU 1 to a Flag Word
T	SW 0..4094	0,8	0,4	Transfer the contents of ACCU 1 to a Special Flag Word
T	DW 0..254	1,6	0,8	Transfer the contents of ACCU 1 to a Data Word
T	PW 0 ..255			Transfer the contents of ACCU 1 to a Peripheral Word
T	ID 0 .. 124	0,7	0,4	Transfer the contents of ACCU 1 to an Input Double Word
T	QD 0..124	0,7	0,4	Transfer the contents of ACCU 1 to an Output Double Word
T	FD 0..252	0,7	0,4	Transfer the contents of ACCU 1 to a Flag Double Word
T	SD 0..4092	0,9	0,5	Transfer the contents of ACCU 1 to a

				Special Double Flag Word
Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
T	DD 0..252	1,8	0,9	Transfer the contents of ACCU 1 to a Data Word
TIR	0 (ACCU1) 2 (ACCU2)	0,8	0,3	Transfer the register contents (addressed by ACCU 1) indirectly into the memory word
TDI	A1 (ACCU1) A2 (ACCU2)	1,0	0,4	Transfer the register contents (addressed by ACCU 1) indirectly into the memory word
TNB	RS 0..255	0,45 /Byte	0,15 /Byte	Transfer a Block Byte (Number 0..255) End address source : ACCU2 (16 Bit) End address target : ACCU1 (16 Bit)
T	RS 0..255	0,7	0,3	Transfer a System Data Word
T=	<Parameter>	1,5	0,7	Transfer the contents of ACCU 1 to the formal Operand (parameter type: BY,W,D)

5.6.5 Timer operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
SP	T 0..255	1,0	0,5	Start a Timer (stored in ACCU 1) as Signal contracting pulse on the leading edge of the RLO
SE	T 0 ..255	1,0	0,5	Start a Timer (stored in ACCU 1) as extended pulse (signal stretching) on the leading edge of the RLO
SR	T 0..255	1,0	0,5	Start an on-delay Timer (stored in ACCU 1) on the leading edge of the RLO
SS	T 0 ..255	1,0	0,5	Start a stored on-delay Timer (stored in ACCU 1) on the leading edge of the RLO
SF	T 0..255	1,0	0,5	Start an off-delay Timer (stored in ACCU 1) on the trailing edge of the RLO
R	T 0 ..255	1,0	0,5	Reset a Timer if RLO= "1"
FR	T 0..255	1,0	0,5	Enable a Timer for cold restart. If RLO="1" "FR T" restarts the Timer
SP=	<Parameter>	2,0	0,9	Start a Timer (formal operand) as a pulse with the value stored in ACCU 1 (FB - Parameter: T)
SEC=	< Parameter >	2,0	0,9	Start a Timer (formal operand) as an extended pulse with the value stored in ACCU 1 (FB - Parameter: T)
SD=	< Parameter >	2,0	0,9	Start an on-delay Timer (formal operand) with the value stored in ACCU 1(FB - Parameter: T)
SSU=	< Parameter >	2,0	0,9	Start a stored on-delay Timer (formal operand) with the value stored in ACCU 1 (FB - Parameter: T)
SFD=	< Parameter >	2,0	0,9	Start an off-delay Timer (formal operand) on the trailing edge of the RLO with the value stored in ACCU1 (FB - Parameter: T)
FR=	< Parameter >	2,0	0,9	Enable formal operand (Timer) for cold restart (FB - Parameter: T)
RD=	< Parameter >	2,0	0,9	Reset the formal operand for a Timer

				(FB parameter: T)

5.6.6 Counter Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
CU	C 0..255	0,9	0,5	Counter counts up 1 on the leading edge of the RLO
CD	C 0..255	0,9	0,5	Counter counts down 1 on the leading edge of the RLO
S	C 0..255	0,9	0,5	Set counter if RLO = "1"
R	C 0..255	0,9	0,5	Reset counter if RLO = "1"
FR	C 0..255	0,9	0,5	Enable a Counter for cold restart. If RLO="1" "FR C" sets, decrements, or increments the Counter.
FR=	< Parameter >	2,0	0,9	Enable formal operand (Counter) for cold restart (FB - Parameter: C)
RD=	< Parameter >	2,0	0,9	Reset the formal operand for a Counter (FB parameter: C)

5.6.7 Arithmetic Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
+F	-	0,6	0,25	ACCU1 = ACCU2 + ACCU1 (16 Bit)
-F	-	0,6	0,25	ACCU1 = ACCU2 - ACCU1 (16 Bit)
xF	-	1,0	0,6	ACCU1 = ACCU2 * ACCU1 (16 Bit)
:F	-	1,5	1,0	ACCU1 = ACCU2 / ACCU1 (16 Bit)
+D	-	0,8	0,35	ACCU1 = ACCU2 + ACCU1 (32 Bit)
-D	-	0,8	0,35	ACCU1 = ACCU2 - ACCU1 (32 Bit)
D	0 .. 255	0,4	0,2	ACCU1 = ACCU2 - Value (8 Bit)
I	0 .. 255	0,4	0,2	ACCU1 = ACCU2 + Value(8 Bit)
ADD	BN -128 .. +127	0,4	0,2	ACCU1 = ACCU2 + BN Value (8 Bit) (Extended Sign)
ADD	KF - 32768..+32767	0,4	0,2	ACCU1 = ACCU2 +/- Value (16 Bit)
ADD	DH 0.. FFFFFFF	0,9	0,4	ACCU1 = ACCU2 + Value (32 Bit)

5.6.8 Comparison Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
!=F	-	0,7	0,3	RLO = ACCU2 = ACCU1 (16 Bit)
><F	-	0,7	0,3	RLO = ACCU2 >< ACCU1 (16 Bit)
>F	-	0,7	0,3	RLO = ACCU2 > ACCU1 (16 Bit)
>=F	-	0,7	0,3	RLO = ACCU2 >= ACCU1 (16 Bit)

<F	-	0,7	0,3	RLO = ACCU2 < ACCU1 (16 Bit)
<=F	-	0,7	0,3	RLO = ACCU2 <= ACCU1 (16 Bit)
!=D	-	0,7	0,3	RLO = ACCU2 = ACCU1 (32 Bit)
><D	-	0,7	0,3	RLO = ACCU2 >< ACCU1 (32 Bit)

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
>D	-	0,7	0,3	RLO = ACCU2 > ACCU1 (32 Bit)
>=D	-	0,7	0,3	RLO = ACCU2 >= ACCU1 (32 Bit)
<D	-	0,7	0,3	RLO = ACCU2 < ACCU1 (32 Bit)
<=D	-	0,7	0,3	RLO = ACCU2 <= ACCU1 (32 Bit)

5.6.9 Block Call Operations

5.6.9.1 Absolute Calls

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
SPA	PB 0 .. 255	2,3	1,3	Jump unconditionally to a Program Block
SPA	FB 0 ..255	2,3	1,3	Jump unconditionally to a Function Block
BA	FX 0 .. 255	2,3	1,3	Jump unconditionally to a Extended Function Block
SPA	SB 0 ..255	2,3	1,3	Jump unconditionally to a Sequence Block
SPA	OB 1 .. 255	2,3	1,3	Call an Organization Block unconditionally
DO=	<Parameter>			Call a Block specified as actual Operands (Process Block Parameter)

5.6.9.2 Conditional Calls (if RLO=1)

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
SPB	PB 0 .. 255	2,6	1,4	Jump conditionally to a Program Block
SPB	FB 0 ..255	2,6	1,4	Jump conditionally to a Function Block
BAB	FX 0 .. 255	2,6	1,4	Jump conditionally to a Extended Function Block
SPB	SB 0 ..255	2,6	1,4	Jump conditionally to a Sequence Block
SPB	OB 1 .. 255	2,6	1,4	Call an Organization Block conditionally

5.6.9.3 Data Block Calls

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
A	DB 1 .. 255			Call a Data Block
AX	DX 0 ..255			Call an Extended Data Block

E	DB 1 .. 255			Generate a Data Block. (No. of Data Block deposited in ACCU 1)
EX	DX 0 ..255			Generate an Extended Data Block. (No. of Data Block deposited in ACCU 1)

5.6.10 Block Return Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
BE	-	2,2	0,9	Block end (termination of a Block)
BEB	-	3,0	1,5	Block end, conditional RLO = 1
BEA	-	2,2	0,9	Block end, unconditional

5.6.11 No Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
NOP	0 .. 1	0,35	0,15	No operation (all bits set / reset)
BLD	0 .. 255	0,5	0,2	Display generation operation

5.6.12 Stop Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
STP	-			Stop: Scanning cycle is not completed. Error ID "STS" is set in the I-Stack.
STS	-			Stop operation. Program processing is interrupted immediately after this operation.

5.6.13 Bit Operations

5.6.13.1 Test a Bit for "1"

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
TB	I 0.0 .. 127.7	1,7	0,9	Test an Input
TB	Q 0.0..127.7	1,7	0,9	Test an Output
TB	F 0.0..255.7	1,7	0,9	Test a Flag
TB	T 0.0..255.15	1,7	0,9	Test a Timer Word Bit
TB	C 0.0..255.15	1,7	0,9	Test a Counter Word Bit
TB	D 0.0..255.15	1,7	0,9	Test a Data Word Bit
TB	RS	1,7	0,9	Test a Data Word Bit in the System data

	0.0..255.15			range

5.6.13.2 Test a Bit for "0"

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
PN	I 0.0 .. 127.7	1,7	0,9	Test an Input
PN	Q 0.0..127.7	1,7	0,9	Test an Output
PN	F 0.0..255.7	1,7	0,9	Test a Flag
PN	T 0.0..255.15	1,7	0,9	Test a Timer Word Bit
PN	C 0.0..255.15	1,7	0,9	Test a Counter Word Bit
PN	D 0.0..255.15	1,7	0,9	Test a Data Word Bit
PN	RS 0.0..255.15	1,7	0,9	Test a Data Word Bit in the System data range

5.6.13.3 Set a Bit

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
SU	I 0.0 .. 127.7	1,7	0,9	Set an Input
SU	Q 0.0..127.7	1,7	0,9	Set an Output
SU	F 0.0..255.7	1,7	0,9	Set a Flag
SU	T 0.0..255.15	1,7	0,9	Set a Timer Word Bit
SU	C 0.0..255.15	1,7	0,9	Set a Counter Word Bit
SU	D 0.0..255.15	1,7	0,9	Set a Data Word Bit
SU	RS 0.0..255.15	1,7	0,9	Set a Data Word Bit in the System data range

5.6.13.4 Reset a Bit

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
RU	I 0.0 .. 127.7	1,7	0,9	Reset an Input
RU	Q 0.0..127.7	1,7	0,9	Reset an Output
RU	F 0.0..255.7	1,7	0,9	Reset a Flag
RU	C 0.0..255.15	1,7	0,9	Reset a Timer Word Bit
RU	Z 0.0..255.15	1,7	0,9	Reset a Counter Word Bit
RU	D 0.0..255.15	1,7	0,9	Reset a Data Word Bit
RU	RS 0.0..255.15	1,7	0,9	Reset a Data Word Bit in the System data range

5.6.14 Conversion Operations

Opera-tion	Operands	Typical execution time in μs		Function
		486 DX2-66	***Pentium 100***	
CFW	-	0,4	0,15	Form the one's complement of ACCU 1 (16 Bit)
CSW	-	0,6	0,3	Form the two's complement of ACCU 1 (16 Bit) CC 1 / CC 0 and OV are affected
CSD	-	0,8	0,4	Form the two's complement of ACCU 1 (32 Bit)
DEF	-	1,7	0,8	Convert a fixed point value into BCD format (16 Bit)
DUF	-	2,3	1,3	Convert a BCD format into a fixed point value (16 Bit)
DED	-	3,5	1,2	Convert a fixed point value into BCD format (32 Bit)
DUD	-	5,5	3,5	Convert a BCD format into a fixed point value (32 Bit)

5.6.15 Shift and Rotation Operations

Opera-tion	Operands	Typical execution time in μs		Function
		486 DX2-66	***Pentium 100***	
SLW	n = 0..15	0,9	0,4	Shift the contents of ACCU 1 to the left by the value n (16 Bit)
SRW	n = 0..15	0,9	0,4	Shift the contents of ACCU 1 to the right by the value n (16 Bit)
SLD	n = 0..32	0,9	0,4	Shift the contents of ACCU (double word) to the left by the value n (32 Bit)
SSW	n = 0..15	0,9	0,4	Shift the contents of ACCU 1 with the sign to the left by the value n (16 Bit)
SSD	n = 0..32	0,9	0,4	Shift the contents of ACCU 1 with the sign to the right by the value n (16 Bit)
RLD	n = 0..32	0,9	0,4	Rotate the contents of ACCU (double word) to the left t by the value n (32 Bit)
RRD	n = 0..32	0,9	0,4	Rotate the contents of ACCU (double word) to the right by the value n (32 Bit)

5.6.16 Jump Operations

Opera-tion	Operands	Typical execution time in μs	Function

		486 DX2-66	***Pentium 100***	
JU=	<Label> max. 4 characters	0,7	0,3	Jump unconditionally to the label.
JC=	<Label> max. 4 characters	0,9	0,5	Jump conditionally (RLO = 1) to the label. (If the RLO is "0", it is set to "1").
JZ=	<Label> max. 4 characters	0,9	0,5	Jump if the result (CC 1 = 0 and CC 0 = 0) is zero. The RLO is not changed.
JN=	<Label> max. 4 characters	0,9	0,5	Jump if the result (CC 1 <> CC 0) is not zero. The RLO is not changed

Jump Operations (continued)

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
JP=	<Label> max. 4 characters	0,9	0,5	Jump if the result (CC 1 = 1 and CC 0 = 0) is > 0. The RLO is not changed.
JM=	<Label> max. 4 characters	0,9	0,5	Jump if the result (CC 1 = 0 and CC 0 = 1) is < 0. The RLO is not changed.
JO=	<Label> max. 4 characters	0,9	0,5	Jump on overflow (OVERFLOW bit set). The RLO is not changed.
JOS=	<Label> max. 4 characters	0,9	0,5	Jump on latching overflow (OVERFLOW bit set). The RLO is not changed.
JUR	-32768..+32767	0,9	0,5	Relative Jump within a Function Block

5.6.17 Other Operations

Opera-tion	Operands	Typical execution time in µs		Function
		486 DX2-66	***Pentium 100***	
IA	-	0,9	0,4	Disable Interrupt
RA	-	6,0	3,0	Enable Interrupt
DO	DW 0 ..255			Process Data Word. The next operation is combined through logic OR with the parameter specified in the Data Word and executed.
DO	FW 0 ..255			Process Flag Word. The next operation is combined through logic OR with the parameter specified in the Data Word and executed.
DI	-			Process via a formal operand (indirectly). The number of the formal operand is in ACCU 1
TAK	-	0,6	0,3	Swap the contents of ACCU 1 and ACCU 2
DO	RS 10 ..			*SoftPLC* Special Instructions

5.6.18 Floating Point operations

The following floating point instructions compatible with the CPU 945 are implemented in the *SoftPLC*45.

5.6.18.1 Convert Operations

FDG	Convert an integer value into a floating point value
GFD	Convert a floating point value into an integer value

5.6.18.2 Load Operations

L KG	Load a constant (floating point value) into ACCU1

5.6.18.3 Comparison Operations

! = G	Compare two floating point values for equal	ACCU2 = ACCU1 ->ROL = ‘1’
> < G	Compare two floating point values for not equal	ACCU2 >< ACCU1 ->ROL = ‘1’
> G	Compare two floating point values for greater	ACCU2 > ACCU1 ->ROL = ‘1’
> = G	Compare two floating point values for greater or equal	ACCU2 > = ACCU1 ->ROL = ‘1’
< = G	Compare two floating point values for less or equal	ACCU2 < = ACCU1 ->ROL = ‘1’
< G	Compare two floating point values for less	ACCU2 < ACCU1 ->ROL = ‘1’

5.6.18.4 Arithmetic Operations

+ G	Add two floating point values	ACCU2 + ACCU1 result ACCU1
- G	Subtract two floating point values	ACCU2 - ACCU1 result ACCU1
* G	Multiply two floating point values	ACCU2 * ACCU1 result ACCU1
/ G	Divide two floating point values	ACCU2 / ACCU1 result ACCU1

The arithmetic operations combine the contents of accumulator 1 and accumulator 2. The result is entered into accumulator 1.

5.6.19 Load and Transfer Operations

Opera-tion	Operand	Function
TNW	RS 0 ... 255 Number of words to transfer	**Transfer a data area** (word wise) (Number 0 .. 255) End address destination: ACCU1 End address source: ACCU2
MBR	= ...F FFFF	**Load the BR Register (Base Address Register)** The BR Register is loaded with a 20 bit constant.
ABR		**Add offset to the BR Register (Base Address Register)** The offset is multiplied by 2 (word offset) and the result is added to the contents of the BR Register. The result is stored in the BR Register.
LRW		**Load the word addressed by the BR Register plus offset** The word (addressed by the BR Register plus offset) is transferred into ACCU1. The previous contents of ACCU1 is transferred to ACCU2. The contents of the BR Register remains unchanged.
LRD	-32768 32767	**Load the double word addressed by the BR Register plus offset** The double word (addressed by the BR Register plus offset) is transferred into ACCU1. The previous contents of ACCU1 is transferred to ACCU2. The contents of the BR Register remains unchanged.
TRW		**Transfer the contents of ACCU1-L into the word addressed by the BR Register plus offset**

		The contents of the ACCU1 (Low word) is transferred into the word (addressed by the BR Register plus offset). The contents of the BR Register remains unchanged.
TRD	-32768 32767	**Transfer the contents of ACCU1 into the double word addressed by the BR Register plus offset** The contents of the ACCU1 is transferred into the word (addressed by the BR Register plus offset). The contents of the BR Register remains unchanged.

5.6.19 Load and Transfer Operations (continued)

Opera-tion	**Operand**	**Function**
MAS		**Transfer the contents of ACCU1 into SAZ (Step Address Counter)** The contents of the ACCU1 (the Bit’s 31 ... 20 of ACCU1 must be zero [0]) are transferred into the Step Address Counter, only Bit 0 to 19 are transferred.
MAB		**Transfer the contents of ACCU1 into BR Register** The contents of the ACCU1 (the Bit’s 31 ... 20 of ACCU1 must be zero [0]) are transferred into the Step Address Counter, only Bit 0 to 19 is transferred.
MSA		**Transfer the contents of the SAZ (Step Address Counter) into ACCU1** The contents of the SAZ are transferred into the ACCU1 (the Bit’s 31 ... 20 of ACCU1 are set to zero [0]) (Address of the next instruction)
MSB		**Transfer the contents of the SAZ (Step Address Counter) into the BR Register** The contents of the SAZ are transferred into the BR Register. (Address of the next instruction)
MBA		**Transfer the contents of the BR Register (Base Address Register) into ACCU1** The contents of the BR Register are transferred into the ACCU1 (the Bit’s 31 ... 20 of ACCU1 are set to zero [0]). Only Bit 0 to 19 is transferred.
MBS		**Transfer the contents of the BR Register into SAZ** The contents of the BR Register is transferred into the SAZ (Step Address Counter)

Attention:

The load and transfer operations do not change the contents of the unused registers.

If the instructions MAS and MBS are used, the address located in the ACCU or the BR Register will point to the instruction to be executed next.

5.6.20 Four Accumulators (CPU 948 compatible)

The *SoftPLC*45 has the possibility to operate with four (4) accumulators like the CPU 948. The system date RS 60 is used to switch the *SoftPLC*45 into the different accumulator modes.

- Four (4) accumulators mode

```
L KF1          ;Load constant (1)
T RS60         ;Transfer into system data RS60
```

- Two (2) accumulators mode

```
L KF0          ;Load constant (0)
T RS60         ;Transfer into system data RS60
```

5.6.20.1 Arithmetic Operations (four accumulators mode)

+ G	Add two floating point values	ACCU2 + ACCU1 result ACCU1
- G	Subtract two floating point values	ACCU2 - ACCU1 result ACCU1
* G	Multiply two floating point values	ACCU2 * ACCU1 result ACCU1
/ G	Divide two floating point values	ACCU2 / ACCU1 result ACCU1

The arithmetic operations combine the contents of accumulator 1 and accumulator 2. The result is entered into accumulator 1.

The accumulator contents are modified according to the table below.

Contents	[ACCU 1]	[ACCU 2]	[ACCU 3]	[ACCU 4]
Prior operation	<ACCU 1>	<ACCU 2>	<ACCU 3>	<ACCU 4>
After operation	<Result>	<ACCU 3>	<ACCU 4>	<ACCU 4>

5.6.20.2 Register Operations (four accumulators mode)

The accumulators 3 and 4 may be used to save intermediate results of complex arithmetic operations. The ENT operation (Enter in arithmetic buffer) is used to load the accumulators 3 and 4.

ENT	Enter in arithmetic buffer ACCU 3 and ACCU 4

The accumulator contents are modified according to the table below.

Contents	[ACCU 1]	[ACCU 2]	[ACCU 3]	[ACCU 4]
Prior ENT operation	<ACCU 1>	<ACCU 2>	<ACCU 3>	<ACCU 4>
After ENT operation	<ACCU 1>	<ACCU 2>	<ACCU 2>	<ACCU 3>

The contents of ACCU 1 and ACCU 2 are unchanged. The original contents of ACCU 4 are lost.

To retrieve the vale from ACCU 4 and ACCU 3 arithmetic operations are used (see chapter 5.6.20.19).

5.7 Memory Allocation of the *SoftPLC* PLC45

The Memory of the *SoftPLC*45 emulates all-important features of the 115U CPU 945 Memory.

Attention:

If accessing the *SoftPLC*45 Memory from a Windows application the Low Byte and the High Byte must be swapped.

Memory Allocation PLC45:

Address	Length Kbytes	Function
0000	4096	S Flags
1000	49152	reserved
D000	2048	reserved
D800	1025	Internal Blocks
DC00	512	OB Address List
DE00	512	FB Address List
E000	512	PB Address List
E200	512	SB Address List
E400	512	DB Address List
E600	512	FX Address List
E800	512	DX Address List
EA00	512	System Data
EC00	256	Timers 0..127
ED00	256	Counters 0..127
EE00	256	Flags
EF00	128	Process Input Image
EF80	128	Process Output Image
F000	256	Timers 128..255
F100	256	Counters 128..255
F200	512	Transfer area for special procedures B RS 28

F400	512	Miscellaneous Values for Timers (reserved)
F600	2560	reserved
10000	720896	Block Buffer

5.8 Address Allocation in the System Data Area

The System Data Area of the *SoftPLC*45 emulates all the important features of the 115U CPU 945 System Data Area.

Unlisted System Data Addresses are not used and reserved for future use.

System Data area structure *SoftPLC*45:

Address (hex.)	**System Data Word**	**Bit**	**Description**
EA08	4	8	Stop at the end of a program execution
EA0A	5	10 11 12 13	Compression aborted Address List Structure Block Shift active Block Shift requested
EA0C	6	2 13 14 15	Alarm enable PLC in Start condition Stop Display Stop Condition from Programming Device *(S5 for Windows*)
EA0E	7	5 6 13	Start not possible Synchronization Failure (Blocks are corrupted) Block Header corrupted
EA10.. EA16	8.. 11		Integrated Clock
EAC0	96		Scan monitoring time (multiple of 10ms)
EAC2	97		Interval timer for OB13 (multiple of 10ms)
EAC4	98		Interval timer for OB12 (multiple of 10ms)
EAC6	99		Interval timer for OB11 (multiple of 10ms)
EAC8	100		Interval timer for OB10 (multiple of 10ms)
EACA	101		Time (in ms) calling OB6
EAF2	121		Actual scan time (in ms)

EAF4	122		Maximum scan time (in ms)
EAF6	123		Minimum scan time (in ms)
EBF0.. EBFF	248.. 255		Reserved for user

The following System Data is only valid if the *SoftPLC*45 is in a Stop condition.

Address (hex.)	System Data Word	Bit	Description
EB00	128		ACCU 1 (High Word)
EB02	129		ACCU 2 (High Word)
EB96	203		ACCU 1 (Low Word)
EB98	204		ACCU 2 (Low Word)
EB9A	205		Statement Register
EB9C	206		Step Address Counter
EB9E	207		Block Stack Pointer
EBA0	208		Start address of the Data Block
EBA2.. EBA8	209.. 212		Bracket Stack (nesting level)
EBAA	213		Interrupt Stack Result Display
		0	First Scan
		1	RLO
		3	OR Memory
		4	Latched Overflow
		5	Overflow
		6	Anz 0 (ACCU 1 Shift)
		7	Anz 1 (ACCU 1 Shift)
EBAC	214		Interrupt Stack Cause of Fault
		2	I/O's not ready
		3	No Warm Start possible. Cold Start requested
		4	Scan time exceeded
		8	Error in the CPU self test routine
		9	Block Stack overflow
		10	Operation interrupted by a programmed Stop
		11	Statement cannot be interpreted
		12	Transfer error for Data Block Statements
		13	Substitution failure

5.9 Organization Blocks integrated in the PLC45

OB 19	Response when calling a not loaded Block
OB 27	Response to substitution errors
OB 31	Scan Time triggering
OB 32	Response to transfer / load errors
OB 250	Operating system service routine
OB 251	PID control algorithm

5.9.1 Response when Calling a not Loaded Block (OB 19)

In OB 19 you can program the response of the *SoftPLC* when a not loaded Block is called. If OB19 is not programmed the *SoftPLC* continues the execution (no response) directly after the jump statement. OB19 may be used to put the *SoftPLC* in the STOP mode when calling a not loaded Block.

5.9.2 Response to Substitution Errors (OB 27)

A substitution error (SUF) can occur when a formal operand has been modified after the PLC program Block was called (JU FBx). When a substitution error occurs, the *SoftPLC* normally goes into the STOP mode. If OB 27 is present the *SoftPLC* is executing the OB 27 instead of going into the STOP mode.

5.9.3 Scan Time triggering (OP 31)

A scan time monitor is integrated. If the cycle time of a program exceeds a preset time (e.g. 600msec), the CPU is forced into the “Stop” mode.

The maximum allowable cycle time can be set in the system data word 96 or in the DB1.

5.9.4 Response to Transfer / Load Errors (OB 32)

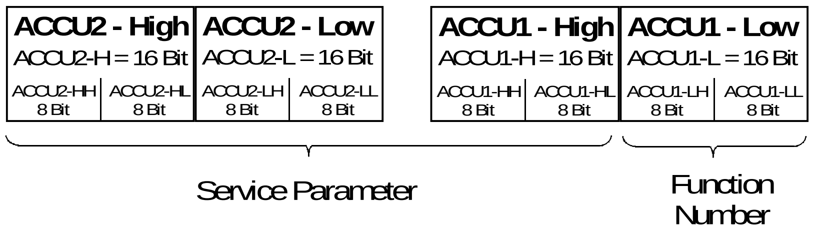
When a transfer / load error (TRAF occurs, the *SoftPLC* normally goes into the STOP mode. If OB 32 is present the *SoftPLC* is executing the OB 32 instead of going into the STOP mode.

5.9.5 Operating System Service Routine (OB 250)

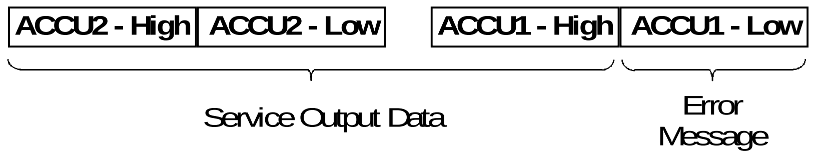
With the operating system service routine, you have the ability to activate various operating system functions or you can modify certain parameters during the cyclic PLC program execution.

A function number is assigned to each operating system service routine. This function number must be loaded into ACCU1 (Low Byte).

The service parameters are loaded into ACCU2 and /or ACCU1 (High Byte)



Error messages are posted to ACCU1 (Low Byte) with the service routine parameters posted to ACCU2 and /or ACCU1 (High Byte).



The following functions are integrated in the *SoftPLC* PLC45:

Operating System Service Routine	Function Number	Parameter	Error Message
Activating OB 6	1	ACCU2-L: 0: Cancel activation 1 ... 65535: Time value (msec)	ACCU1-L: 0: no Error
New Interval for OB 10	2	ACCU2-L: 0: no OB 10 call 1 ... 65535: Time value (msec)	ACCU1-L: 0: no Error
New Interval for OB 11	3	ACCU2-L: 0: no OB 11 call 1 ... 65535 Time value (msec)	ACCU1-L: 0: no Error
New Interval for OB 12	4	ACCU2-L: 0: no OB 12 call 1 ... 65535 Time value (msec)	ACCU1-L: 0: no Error
New Interval for OB 13	5	ACCU2-L: 0: no OB 13 call 1 ... 65535 Time value (msec)	ACCU1-L: 0: no Error
Generating a DB without TRAF	10	ACCU2-LL: 0 ... 255: DB Number ACCU2-H: 0: delete DB	ACCU1-L: 0: no Error 5: TRF (identical

		1 ... FFF9: DB size until including DW	to EDB)
Generating a DX without TRAF	11	ACCU2-LL: 0 ... 255: DX Number ACCU2-H: 0: delete DX 1 ... FFF9: DX size until including DW	ACCU1-L: 0: no Error 5: TRF (identical to EXDX)
Address (Byte) reading from the S5 Bus	13	ACCU2-L: 0 ... FFFF: Address	ACCU1-L: 0: no Error 5: QVZ ACCU2-LL Byte read

The following functions are integrated in the *SoftPLC* PLC45: (continued)

Operating System Service Routine	**Function Number**	**Parameter**	**Error Message**
Address (Byte) writing to the S5 Bus	14	ACCU2-L: 0 ... FFFF: Address ACCU2-HL: 0 ... FF: Byte to be written	ACCU1-L: 0: no Error 5: QVZ
Address (Word) reading from the S5 Bus	15	ACCU2-L: 0 ... FFFF: Address	ACCU1-L: 0: no Error 5: QVZ 7: invalid Address ACCU2-L Word read
Address (Word) writing to the S5 Bus	16	ACCU2-L: 0 ... FFFF: Address ACCU2-HL: 0 ... FF: Word to be written	ACCU1-L: 0: no Error 5: QVZ 7: invalid Address

5.9.6 PID control algorithm (OB 251)

The *SoftPLC* has an integrated PID control algorithm (OB 251). The execution time in the *SoftPLC* is about 100 times faster than the execution time in a hardware PLC. To enhance the accuracy of the PID control algorithm the *SoftPLC* uses ***REAL*** values for the calculations. Prior to calling the OB251 the PID control Data Block (DB) holding the controller parameters and other controller specific data must be open. The PID control algorithm is called periodically (sampling interval) and calculates the new variables.

6 DDE Manager

The *SoftPLC* supports the **D**ynamic **D**ata **E**xchange (DDE) to exchange data between *SoftPLC* and other Windows applications.

During the installation, an icon for the **PLC43/45 DDE Manager** is added to the program group, *S5 for Windows*. The name of the DDE Manager Program is **PLC43DDE.EXE.** The file name is always **PLC43DDE.EXE** even when you install the *SoftPLC*45.

If you need constant access to the *SoftPLC* via DDE you should put the icon into the Auto Start group.

The DDE Manager is completely independent from *S5 for Windows.* Its functionality however is compatible.

The procedure to exchange data between the *SoftPLC* and your Windows application, via the DDE, depends on your application. Please refer to the user manual for information on how to setup the data exchange with another Windows application (*SoftPLC*).

The DDE Manager is a DDE Server with the following functions:

- Request ()
- Poke ()
- Advise ()
- Unadvise ()
- Coldlink and Hotlink

With the function, *Request*, a value may be read from the *SoftPLC*. With the function, *Poke*, a value may be written into the *SoftPLC*. To establish a *"Hotlink"* connection between the *SoftPLC* and your Windows application the function, *Advice*, is used. *Unadvise* cancels this link.

The Server Name of the DDE Manager is **'PLC43'**. The Topic Name is **'PLC'**. The Server Name is always **'PLC43'** even when you install the *SoftPLC*45.

Name format of the data to be exchanged with the *SoftPLC* :

Value		Name Format	Example
Input Bit	In	En.n	E1.1
Input Byte	IBn	En or Ebn	E1 or EB3
Input Word	Wn	Ewn	EW123
Output Bit	Qn	An.n	A1.1
Output Byte	QBn	An oder Abn	A1 or AB3
Output Word	QWn	Awn	AW123

Name format of the data to be exchanged with the *SoftPLC* (continued):

Value	**Name Format**	**Example**
Flag Bit Fn	Mn.n	M1.1
Flag Byte FYn	Mn oder MBn	M1 or MB3
Flag Word FWn	MWn	MW123
S Flag Bit SFn	Sn.n	S1.1
S Flag Byte SFBn	Sn oder SYn	S1 or SY3
S Flag Word SFWn	SWn	SW123
Data Word DWn, dbn	DWn,dbn	DW3,12 (DW3 of DB12)
Data Byte left DLn, dbn	DLn,dbn	DL3,12 (DL3 of DB12)
Data Byte right DRn, dbn	DRn,dbn	DR3,12 (DR3 of DB12)
Data Word from DX DWn, Xdbn	DWn,Xdbn	DW3,X12 (DW3 of DX12)
Data Bytes left from DX DLn, Xdbn	DLn,Xdbn	DL3,X12 (DL3 of DX12)
Data Bytes right from DX DRn, Xdbn	DRn,Xdbn	DR3,X12 (DR3 of DX12)
Timer T	Tn	T1
Counter C	Zn	Z5

6.1 Example

With the Windows application Microsoft EXCEL, the Flag Byte 10 (FY10) from the *SoftPLC* should be read and displayed.

In one of the Cells write the following Instruction:

=PLC43|PLC!MB10

This instruction is a Remote Reference Formula. This formula automatically establishes a *"Hotlink"* to the *SoftPLC*. Whenever the value of the Flag Byte FB10 changes, the new value is transmitted and displayed in EXCEL. Due to different versions of Microsoft EXEL it is possible that the Remote Reference Formula (each part must be in **' '**) must be entered as followed:

='PLC43'|'PLC'!'MB10'

Loading the Flag Bit F 3.2 from Microsoft EXCEL into the *SoftPLC*.

Generating a Macro using Microsoft EXCEL

- Menu Insert
- Instruction Macro
- Instruction Module. The Module Worksheet opens.

Insert the following Macro in Visual Basic:

```
Sub WriteToPlc(Operand, Value)

DIM SendValue As Object

Set SendValue = Cells(11, 2)

SendValue = "S5VAL" + CStr(Value)

DDEChannel = DDEInitiate("PLC43"; "PLC")

DDEPoke DDEChannel, Operand, SendValue

DDETerminate (DDEChannel)

EndSub


Sub Manual()

WriteToPlc "M3.2"; 1

End Sub
```

To start the macro "Manual" you have to create a Button in Sheet 1.

- ◆ Open Sheet 1.
- ◆ Click the Drawing Icon to display the drawing toolbar.
- ◆ From the drawing toolbar click the Create Button icon.
- ◆ Draw a Button using the mouse.
- ◆ When the Assign Macro dialog box opens, assign "Manual" to the button.
- ◆ Close the dialog box and save the Sheet. Now you may start the macro by pressing the Button.

The execution of the macro "Manual" writes a logical 1 (one) into the Flag Byte F3.2 of the *SoftPLC*.

The String "S5VAL=" in front of the SendValue expedites the execution with EXCEL. With other Windows application this String is not needed.

7 Visualization

Several visualization software manufactures have drivers for their visualization software package available for fast direct access of the *SoftPLC*

7.1 Fast Direct Access with Visual Basic

With Visual Basic version 4.0 you can access the *SoftPLC* directly. The required DLL is part of the *SoftPLC* software package (PLC32.DLL).

You will find an example of how to call the function in the directory \DLL\VB4 on your hard disk after installing the *SoftPLC* software. Details on how to use the example are explained in the README.TXT file.

Appendix

The Key Program

The *SoftPLC* software can only be used if the **Key** to unlock the program is present in the *SoftPLC* directory (**PLC43** default).

The key to unlock the *SoftPLC* software is transferred to the *SoftPLC* directory during the installation process.

Whenever you want to move the *SoftPLC* software to an other hard disk, you must first return the key to the **Key Disk**.

The **Key Program** is used to return the **Key** from the *SoftPLC* directory to the key disk. You may also use other functions of this program to transfer the key manually from the key disk to a *SoftPLC* directory to unlock the software. The key program also provides a tool to modify a key to allow the installation of additional options (if purchased at a later date).

The **Key Program** is a series of DOS programs and must be executed from the DOS Window.

The next chapters explain how to use the **Key Programs**.

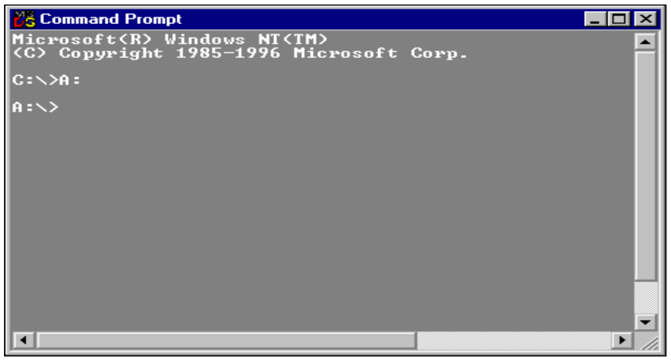
DOS Key Utilities

The following DOS Utilities are on the key disk.

- CCLOOK.EXE
- CCMOVE.EXE
- CCCHANGE.EXE

To execute any of these programs open the DOS Window.

- Insert the key disk in the proper floppy disk drive.
- Make the floppy drive active by typing **a:** or **b:**, depending on where you inserted the key disk.



```
Microsoft(R) Windows NT(TM)
(C) Copyright 1985-1996 Microsoft Corp.

C:\>A:

A:\>
```

Figure 23

CCLOOK

This program displays the status of the key disk or the status of the *SoftPLC* directory.

Use the following syntax to display the status.

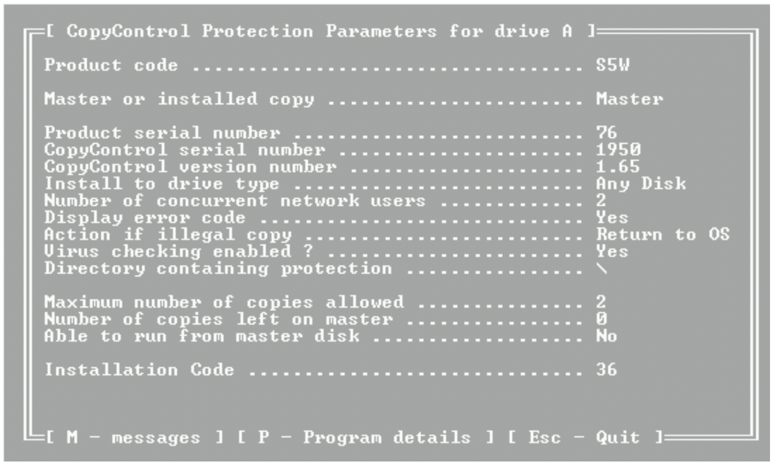
CCLOOK < path >

To run **CCLOOK** do the following steps.

Make sure that the drive where you inserted the key disk is active and the key disk is not write protected.

◆ Type ***cclook***. The status display appears.

Example of the status display of a key disk.



```
[ CopyControl Protection Parameters for drive A ]

Product code .................................... S5W

Master or installed copy ........................ Master

Product serial number ........................... 76
CopyControl serial number ....................... 1950
CopyControl version number ...................... 1.65
Install to drive type ........................... Any Disk
Number of concurrent network users .............. 2
Display error code .............................. Yes
Action if illegal copy .......................... Return to OS
Virus checking enabled ? ........................ Yes
Directory containing protection ................. \

Maximum number of copies allowed ................ 2
Number of copies left on master ................. 0
Able to run from master disk .................... No

Installation Code ............................... 36


[ M - messages ] [ P - Program details ] [ Esc - Quit ]
```

Figure 24

The important information concerning the key disk status is.

- Product serial number
- Maximum number of copies allowed
- Number of copies left on master
- Number of concurrent network users

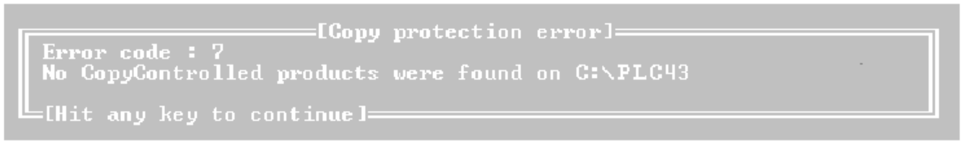
To display the status of the *SoftPLC* directory use the following syntax.

CCLOOK [drive: *SoftPLC* directory]

Example:

◆ Type ***cclook c:\PLC43*** The status display appears.

If the CCLOOK program cannot find an unlocked *SoftPLC* directory an error message is displayed.



```
[Copy protection error]
Error code : 7
No CopyControlled products were found on C:\PLC43

[Hit any key to continue]
```

Figure 25

CCMOVE

This program is used to transfer the key, to and from, the key disk and the *SoftPLC* directory.

Use the following syntax to move the key.

CCMOVE < from: > <to:>

< from: >	is the source path
<to:>	is the destination path

To run **CCMOVE** do the following steps.

Make sure that the drive where you inserted the key disk is active and the key disk is not write protected.

Returning the Key

To return the key **from** the *SoftPLC* directory **to** the key disk, do the following (assuming the path of the *SoftPLC* directory is **C:\PLC43** and the drive with the key disk is drive **A:**).

- Type ***ccmove a: c:\ PLC43*** The key is transferred to the *SoftPLC* directory to unlock the software.

Transferring the Key.

To transfer the key **from** the key disk **to** the *SoftPLC* directory, do the following (assuming the path of the *SoftPLC* directory is **C:\PLC43** and the drive with the key disk is drive **A:**).

- Type ***ccmove a: c:\PLC43*** The key is transferred to the *SoftPLC* directory to unlock the software.

CCCHANGE

This program is used to modify the original key stored on the key.

Use the following syntax to change the key.

CCCHANGE < path >

Example:

Make sure that the drive where you inserted the key disk is active and the key disk is not write protected.

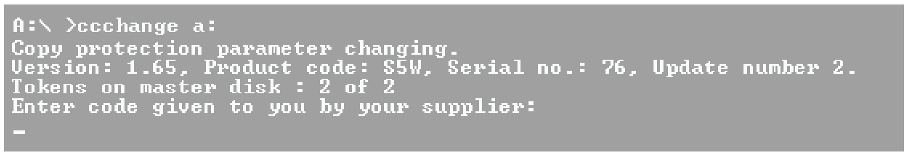
Assuming the drive with the key disk is drive **A:**

◆ Type ***ccchange***

Information about the CCCHANGE program will be displayed. Here you will find the version of software (example 1.65), the serial number (example 76) and the update number (example 2).

The serial number, update number and version number are required by the hot line engineer to assist you.

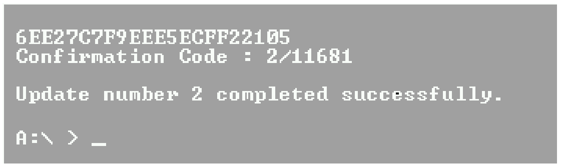
Example of the prompting when running CCCHANGE



```
A:\ >ccchange a:
Copy protection parameter changing.
Version: 1.65, Product code: S5W, Serial no.: 76, Update number 2.
Tokens on master disk : 2 of 2
Enter code given to you by your supplier:
_
```

Figure 26

Enter the key word (code number) and confirm with the **È** key. The completion of the modification will be confirmed with the following text



```
6EE27C7F9EEE5ECFF22105
Confirmation Code : 2/11681

Update number 2 completed successfully.

A:\ > _
```

Figure 27

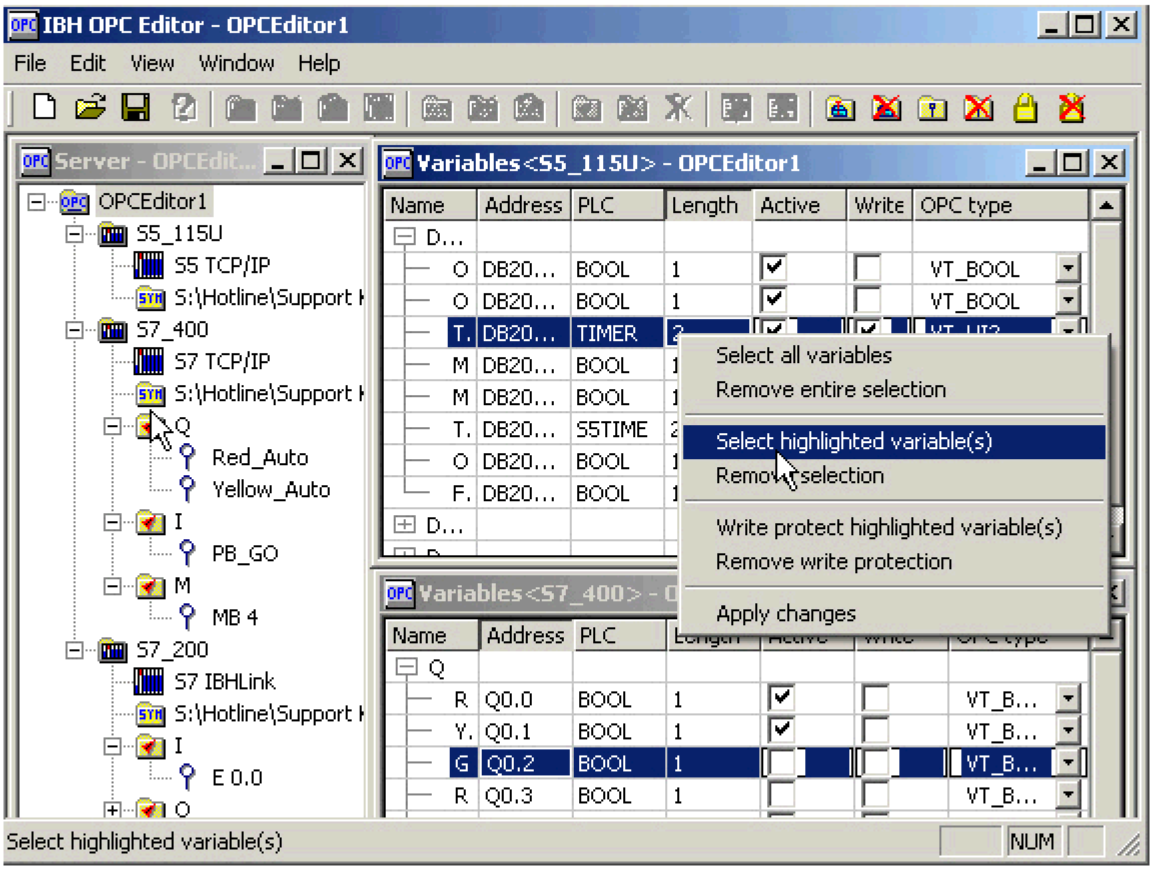
The modified key disk is now ready for further use.

8 Null Modem Cable

Sub.-D Connector			**Sub.-D Connector**	
Female			**Female**	
9 Pin	**25 Pin**		**9 Pin**	**25 Pin**
1	------	Not Connected	1	------
2	3	-------------------------------	3	2
3	2	-------------------------------	2	3
4	20	-------------------------------	6	6
5	7	-------------------------------	5	7
6	6	-------------------------------	4	20
7	4	-------------------------------	8	5
8	5	-------------------------------	7	4
9	------	---- Not Connected ----	9	------
Shell	Shell	---------- Shield --------	Shell	Shell

IBH OPC Server

With the IBH OPC Server you can link a Visualization application with a Simatic® PLC S5, S7-200,S7300 and S7-400 or an IBHsoftec SoftPLC. Also a mixed operation is possible.



Name	Address	PLC	Length	Active	Write	OPC type
D...						
O	DB20...	BOOL	1	✓	☐	VT_BOOL
O	DB20...	BOOL	1	✓	☐	VT_BOOL
T.	DB20...	TIMER	2	✓	✓	VT_UI2
M	DB20...	BOOL	1			
M	DB20...	BOOL	1			
T.	DB20...	S5TIME	2			
O	DB20...	BOOL	1			
F.	DB20...	BOOL	1			
D...						

Name	Address	PLC	Length	Active	Write	OPC type
Q						
R	Q0.0	BOOL	1	✓	☐	VT_B...
Y.	Q0.1	BOOL	1	✓	☐	VT_B...
G	Q0.2	BOOL	1	☐	☐	VT_B...
R	Q0.3	BOOL	1	☐	☐	VT_B...

Access to the variables of a PLC Control via OPC. The symbolic addressing used within the PLC program and the Data Blocks can directly be used within the HMI.
With a few mouse clicks all or only the desired variables from the PLC can be selected.
The following file formats are supported: S5 for Windows®, S7 for Windows®, STEP®5 and STEP®7.