MAC Operator Terminal

Human-Machine-Interface

User's Manual

English/Svensk

PROFIBUS DP
Communication Module IFC PBDP

MITSUBISHI ELECTRIC EUROPE B.V.
FACTORY AUTOMATION

ENGLISH

Manual PROFIBUS DP

Foreword

This manual is an installation and function description for the PROFIBUS DP communication module IFC PBDP.
The module is connected to the E300, E700 and E710 terminals.
Besides this manual, the following manuals are also available.

- E700, Installation
- E710, Installation
- E700/E710, Manual

For information about Siemens PLC systems, refer to respective manuals.

Safety precautions

General

- Check the delivery for transport damage. If damage is found, advise your supplier.
- The product fulfils the requirements of article 4 of EMC directive 89/336/EEC.
- Do not use the product in an explosive environment.
- Modifications, changes and additions to the product are forbidden.
- Use only spare parts approved by Mitsubishi Electric Europe B.V.
- Read the user instructions carefully before use.
- This equipment should only be operated by qualified personnel.

At installation

- The product is constructed for stationary installation.
- Install the product according to the accompanying installation instructions.
- The product must be grounded according to the accompanying installation instructions.
- This equipment must be installed by qualified personnel.
- High voltage-, signal- and supply cables must be separated.
- The product should not be mounted in direct sunlight.

In use

- Keep the equipment clean.
- Emergency stop- and other safety functions should not be controlled from the terminal.
- Do not touch the keys, displays, etc. with sharp objects.

Service and maintenance

– The agreed guarantee applies.
– Clean the display and face with a soft cloth and mild detergent.
– Use batteries specified by G & L Beijer Electronics AB. Batteries should be changed by qualified personnel. The person changing the batteries should be grounded during the operation; e.g. with a grounded wrist strap.
– Repairs should be made by qualified personnel.

At disassembly and scrapping

– Local regulations apply concerning recycling of products or part.
– Please note that the electrolyte condenser and display contain hazardous substances.

Contents

1 Introduction

PROFIBUS DP is a vendor-independent, open industrial fieldbus which can be used in a very wide range of applications. It is an established technology with a large installed base.

Process automation equipment such as sensors, actuators, transmitters, drives and programmable logic controllers, increasingly use digital microelectronics.

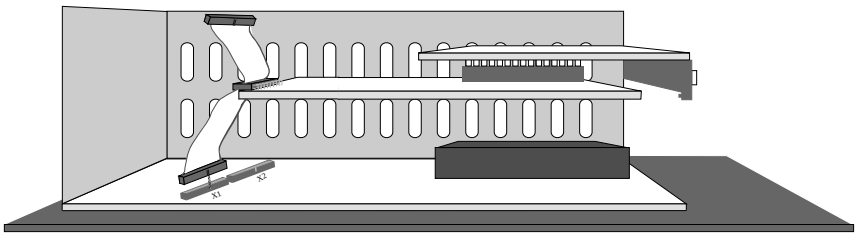
PROFIBUS ensures that devices from different vendors can communicate together without the need to adapt interfaces. PROFIBUS is standardized as European Standard pr EN 50170.

The PROFIBUS DP card is dupplied with a type diskette containing PLC programs for communication with Mitsubishi Electric PLC system MELSEC A and Siemens PLC system SIMATIC S5 and SIMATIC S7.

2 Installation

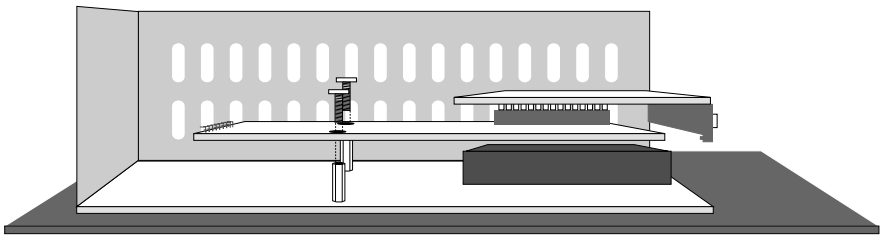
This chapter describes how the IFC PBDP card is connected to the terminal. The IFC PBDP packet includes the IFC PBDP card and a type diskette.

2.1 How to connect the flat cable



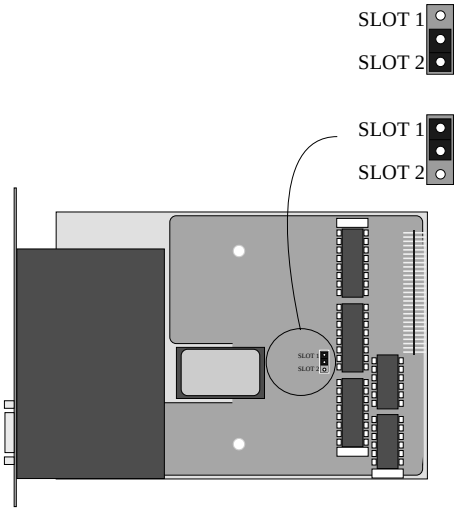
1. Unscrew the back panel and the slot panel.
2. Connect the flat cable to the connector X1.

2.2 How to connect the IFC PBDP card



Mount the IFC PBDP card using the enclosed spacers and screws.

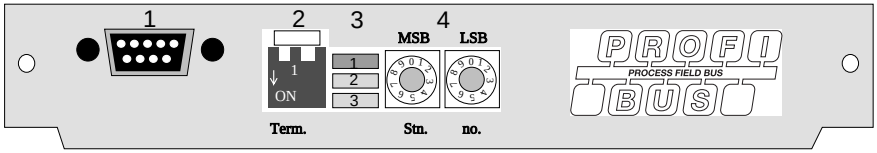
2.3 How to select the physical port



Slot 2 is selected

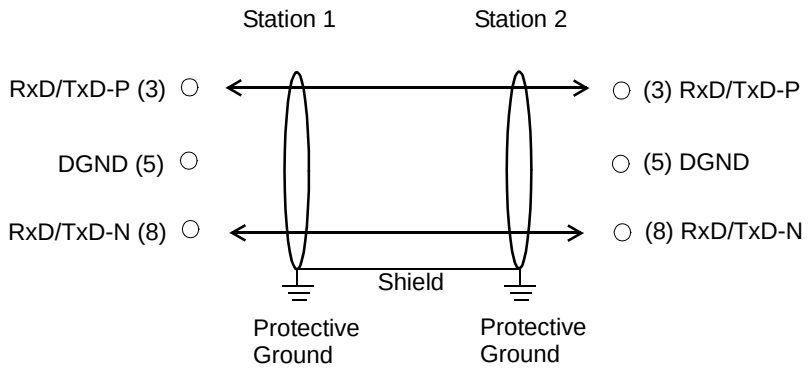
Slot 1 is selected
(For E300 you have to
select slot 1.)

2.4 Communication settings for the IFC PBDP card



1. Connector for the communication cable.
2. Bus termination. Set to ON on the first and last units in the network. The first unit in the network often is the master unit in the PLC system.
3. 1: Red, **ERR**, Configuration or communication error. The LED is red until the unit is configured, Indicates time out.
   2: Green, **PWR**, Power supply 5 VDC OK.
   3: Green, **DIA**, Diagnostic error, not used.
4. State the station number.

2.5 Cabel to PROFIBUS-DP



2.6 Technical data

I/O area size	32-200 byte
Baudrate	9600 bit/s - 12 MBit/s
Identity code	1002
Max. number of nodes without repeater	32
Max. number of nodes with repeater	96
Max. cable length (without repeater)	3000m, 9.6 kb
Max. cable length (without repeater)	200m, 12 Mb

The cable Unitronic-Bus L2/F.I.P is tested and has the following performance:

Capacitance	30 nF/km
Impedance	150 Ohm (3-20 MHz)
Resistance	115 Ohm/km

3 Configure the terminal

The configuration is done with CIMREX PROG.

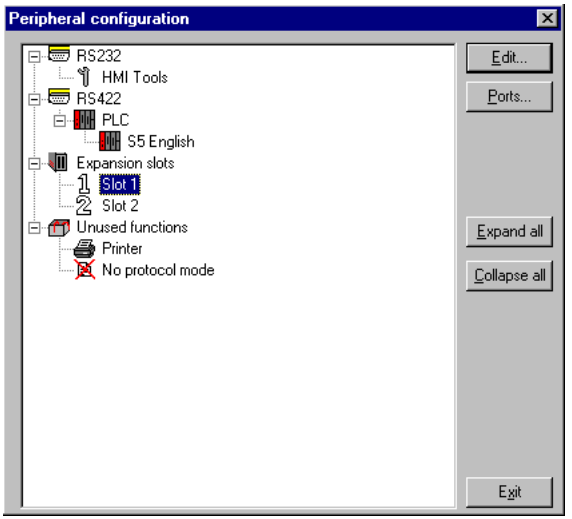
Make the following steps to use a PCMCIA Flash memory card.

1. Install the IFC MC card in the terminal.
2. Connect the Flash memory card in the IFC MC-module.

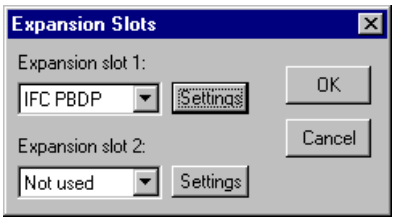
3.1 Define slot

1. Select **Peripherals** in the **Setup** menu.

2. Select the slot you define with the jumper J1 on the expansion card and press **Edit...**.



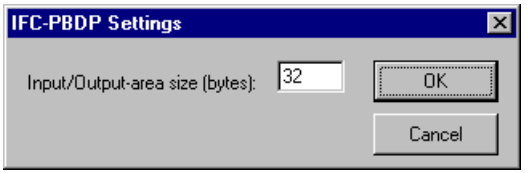
3. Select type of expansion card, in this case IFC PBDP.



4. Select Settings. Here you select the function the memory card will be formatted for.

Settings:

State the settings for the MMI profile.



Input/Output-area size (bytes):

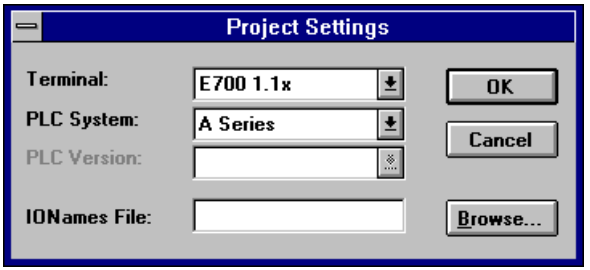
State the size of the input and output area in bytes. Default setting is 32 bytes.

4 Connection to MELSEC A

With the PROFIBUS DP card, the MMI terminal can communicate with the Mitsubishi A(1S)J71PB92 module. For information about the A series PLC system we refer to the respective manual.

4.1 Selection of PLC system

In the dialog Project Settings in MAC Programmer+ you select which PLC system the MMI terminal should be connected to. Select A Series.



4.2 I/O handling

The terminal can handle the following data types in MELSEC A.

Data types MELSEC	Data types IEC	Description
X	**%IX**	Input
Y	**%QX**	Output
M	**%MX**	Memory cell
B	**%MX**	Link memory cell (MELSEC NET)
D	**%MW**	Data register
W	**%MW**	Link register (MELSEC NET)
R	**%MW**	File register
T	**%MW**	Timer, current value
C	**%MW**	Counter, current value

Note!
The MMI profile uses a number of memory cells and data registers for internal handling. These can not be used for other purposes in the PLC program.

Digital signals

Data type MELSEC	Data type IEC
Xn	**%IX**b
Yn	**%QX**b
Mn	**%MX0.**b
Bn	**%MX1.**b

n=address, b=bit number

Analog signals

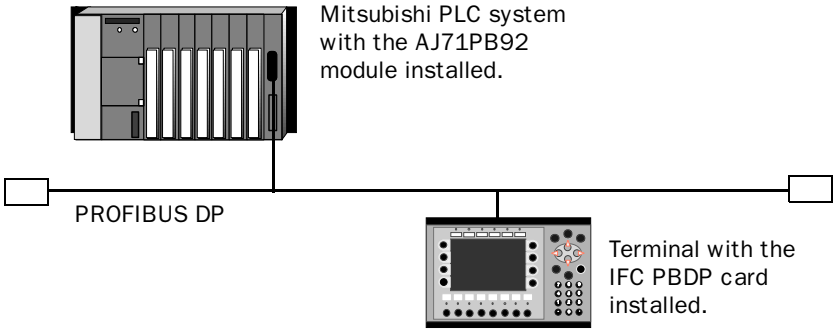
Data type MELSEC	Data type IEC
Dn	**%MW0.**n
Wn	**%MW1.**n
Rn	**%MW2.**n
Tn	**%MW3.**n
Cn	**%MW4.**n

n=address

Refer to the respective manual for more information about addressing to Mitsubishi Electric PLC system MELSEC A.

4.3 Example

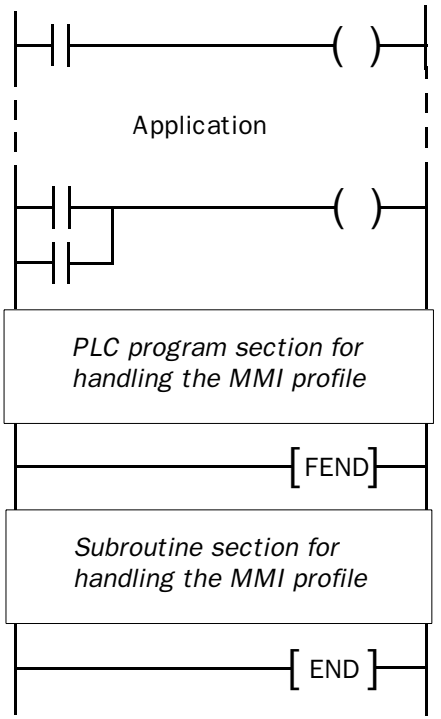
In this example we use a Mitsubishi MELSEC A system with the module AJ71PB92 and the PLC program delivered on the typefile diskette together with the IFC PBDP card. The example describes the sequence for making settings and connections to get correct communiction.



1. Install the terminal according to the Installation manual delivered with the terminal.
2. Configure the terminal via the software package MAC Programmer+. The settings to the IFC PBDP card are made in the **Setup** menu under **Expansion slots**. See the chapter *Configure the MMI terminal*.
3. Start the PROFIBUS configuration software e.g. COM ET200.
4. Configure the master, baudrate, station number, number of bytes in the transmission area, etc. See the software manual.Type files for the terminal are available on the IFC PBDP diskette.
5. Create a binary file. For information see the software manual for the COM ET200 for DOS.
6. Start the program SWOIX-DPLD from DOS. Load the binary file to the AJ71PB92 module via the SWOIX-DPLD program. For more information see the manual for SWOIX-DPLD.
7. Load the enclosed PLC programs to the A series CPU via MELSEC MEDOC.

8. Connect the cable between the AJ71PB92 module in the PLC system and the IFC-PBDP card in the terminal.
9. Put the PLC system and MMI terminal in run mode.

The following figure shows how to implement the program section and subroutine section in the PLC program.



Printout of PLC program section

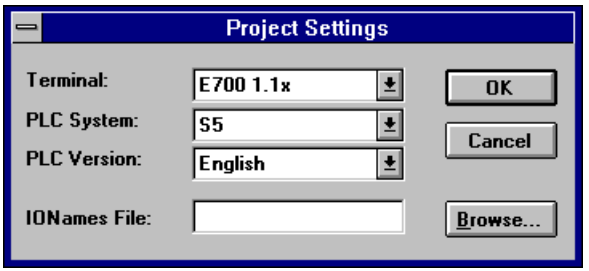
In the chapter *Appendix for printouts* at the back of the manual you will find a printout of the PLC program section PROFILE for PROFIBUS DP communication.

5 Connection to SIMATIC S5

With the PROFIBUS DP card the MMI-terminal can communicate with Siemens PLC system SIMATIC S5. This chapter describes how the function blocks are built up, which I/O the terminal can access, and how the PLC program is called. For more information about SIMATIC S5, please refer to the Siemens documentation for the SIMATIC S5.

5.1 Selection of PLC system

I the dialog Project Settings in MAC Programmer+ you select which PLC system the MMI terminal should be connected to. Select S5.



5.2 I/O handling

The terminal can handle the following data types in SIMATIC S5:

Data type English	Data type German	Description
F	**M**	Flag
Q	**A**	Output
I	**E**	Input
DB	**DB**	Datablock

The DB (Data block) in SIMATIC S5 can have a maximum length of 256 words. The terminal can access all DBs in the PLC system.

All data types consists of byte areas. Addressing is always byte-specific, regardless of whether it is 1, 16 or 32 bits. The addresses are always decimal, 0-65535.

For information about instructions in SIMATIC S5, refer to the SIMATIC manual.

Digital signals

For digital signals, you state current bit in the byte. For example, I50.3 means bit 3 in input byte 50.

Data type English	Data type German
Ixxxxx.b	**E**xxxxx.b
Qxxxxx.b	**A**xxxxx.b
Fxxxxx.b	**M**xxxxx.b

xxxxx=address 0-65535, b=bit number 0-7

Analog signals

For 16-bit numbers, you state the suffix W after the data type; e.g. MW100 means 2 bytes from memory byte 100-101.

Data type English	Data type German
IWxxxxx	**EW**xxxxx
QWxxxxx	**AW**xxxxx
FWxxxxx	**MW**xxxxx
DBno.**DW**adr	**DB**no.**DW**adr

xxxxx=address 0-65535, no=database number 0-255 and adr=data word within the data base 0-255.

5.3 Description of the PLC program section

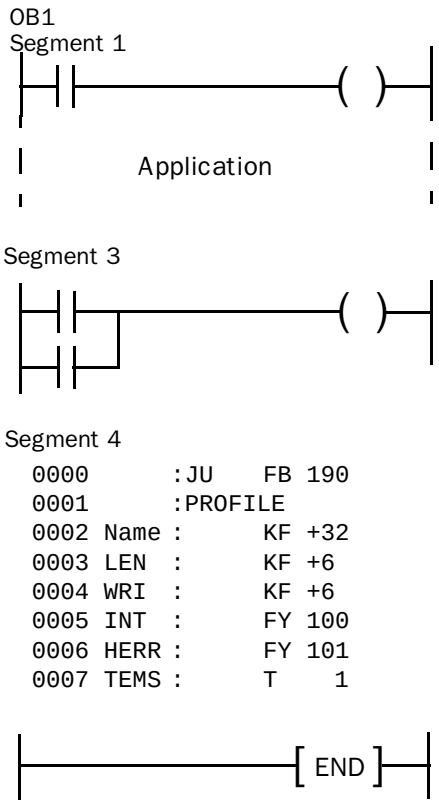
The PLC program section consists of three function blocks plus one block (OB1) which addresses the function block 190.

Program block

The PLC program on the type diskette consists of three function blocks and one main program.

Function block	Description
OB1	Main program. Calls the function block 190.
FB 190	This block is called by the OB1 and handles the MMI profile.
FB 191	This block reads one index.
FB 192	This block writes one index.

The following figure shows how to implement OB1 in the PLC program.



The main program, OB1

OB1 is the main program where the parameters are defined for calling the other function blocks.

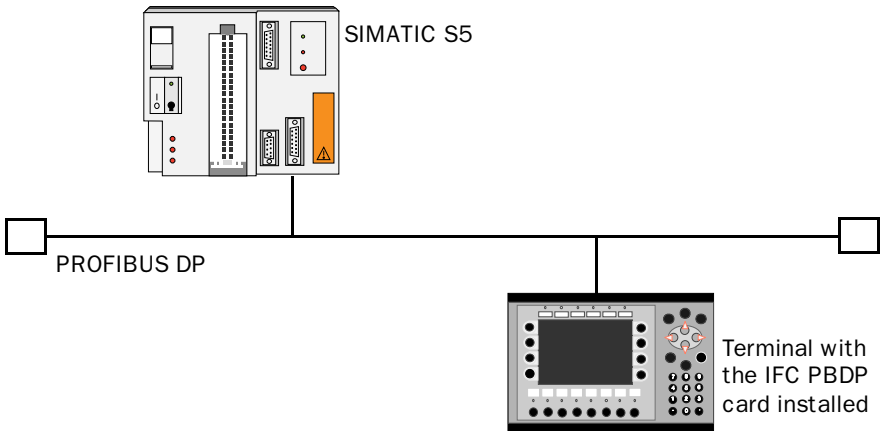
The following parameters are defined for the function block 190:

Parameter	Description
LEN	State the length of the request container and response container. Must be the same as the setting in the terminal.
READ	State the address to the first byte in the response container in the PROFIBUS area.
WRI	State the address to the first byte in the request container in the PROFIBUS area.
INT	Internal byte
HERR	State the register to contain eventual error code from FB 190.
TMS	Time out in seconds. If communication with the terminal brakes for a longer time than stated in the parameter TMS an error code will be set in the parameter HERR.

For more information about the parameters, refer to the manual for SIMATIC S5.

Example

In this example we use Siemens PLC system S5 and the PLC program on the type diskette. The example describes in which order you make the settings and connections to get the correct communication.



1. Install the terminal according to the Installation manual delivered with the terminal.
2. Configure the terminal with the software package MAC Programmer+. The settings for the IFC PBDP card is made in the **Setup** menu under **Expansion slots**.
3. Start the COM ET200 configuration software.
4. Configure the master, baudrate, station number, number of bytes in the transfer container etc. For more information, refer to the software manual. Type files for the terminal are available on the IFC PBDP diskette.
5. Load the configuration to the S5. See the SIMATIC S5 manual.
6. Load the enclosed PLC program to the S5.
7. Connect the cable between the S5 system and the IFC PBDP card in the terminal.

8. Put the PLC system and the MMI terminal in run mode.

Note!
If you try to open a non-existent data block the PLC system will stop. For more information, refer to the SIMATIC manual.

Printout of PLC program section

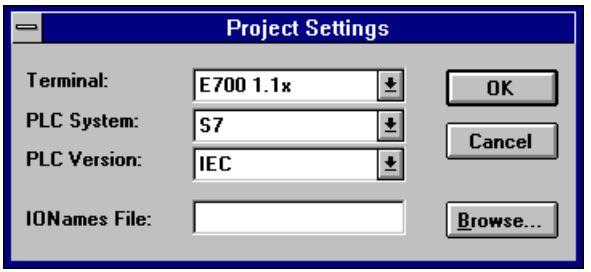
In the chapter *Appendix for printouts* at the back of the manual you will find a printout of the PROFIBUS DP function block.

6 Connection to SIMATIC S7

With the PROFIBUS DP card, the MMI-terminals E700 and E710 can communicate with Siemens PLC system SIMATIC S7. This chapter describes how the function blocks are built up, which I/O E700 and E710 can access, and how the PLC program is addressed. For more information about S7, refer to the Siemens documentation for the SIMATIC S7.

6.1 Selection of PLC system

In the diglog Project Settings in MAC Programmer+ you select which PLC system the MMI terminal should be connected to. Select S7.



6.2 I/O handling

The E700/E710 can handle the following data types in SIMATIC S7:

Data types IEC	Data types German	Description
M	**M**	Flag
Q	**A**	Output
I	**E**	Input
DB	**DB**	Data block

The project memory decides the max length of the DB (Datablock) in SIMATIC S7. The terminal can access one DB in the PLC system.

All data types consists of byte areas. Addressing is always byte-specific, regardless of whether it is 1, 16 or 32 bits. The addresses are always decimal, 0-65535.

For information about the instructions in S7 we refer to the SIMATIC manual.

Digital signals

For digital signals, you state current bit in the byte. For example I50.3 means bit 3 in input byte 50.

Data type IEC	Data type SIMATIC
Ixxxxx.b	**E**xxxxx.b
Qxxxxx.b	**A**xxxxx.b
Mxxxxx.b	**M**xxxxx.b

xxxxx=address 0-65535, b=bit number 0-7

Analog signals

For 16-bit numbers, you state the suffix W after the data type; e.g. MW100 means 2 bytes from memory byte 100-101.

Data type IEC	Data type SIMATIC
IWxxxxx	**EW**xxxxx
QWxxxxx	**AW**xxxxx
MWxxxxx	**MW**xxxxx
DBWxxxxx	**DBW**xxxxx

xxxxx=address 0-65535.

6.3 Description of the PLC program section

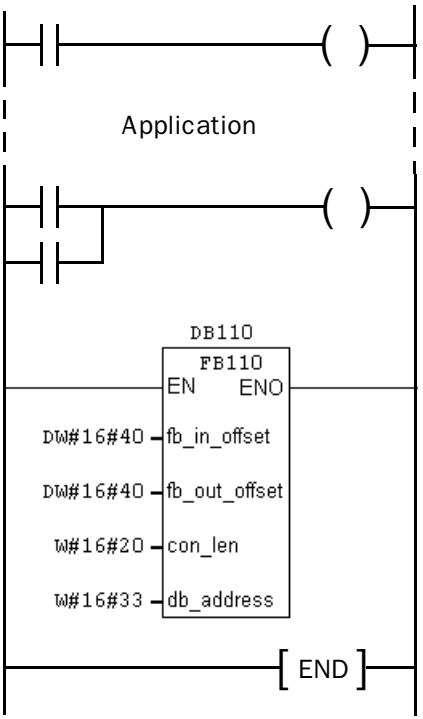
The PLC program consists of one function block and a main program (OB1) which addresses the function block.

Program block

The PLC program on the type diskette consists of three function blocks and one main program.

Function block	Description
OB1	Main program. Calls the function block 110.
FB 110	This block is called by the OB1 and handles the MMI profile.
FC 111	This block reads one index.
FC 112	This block writes one index.
DB 51	Data block used for analog signals.

The following figure shows how to implement OB1 in the PLC program.



The main program, OB1

OB1 is the main program where parameters are defined for calling the function block 110.

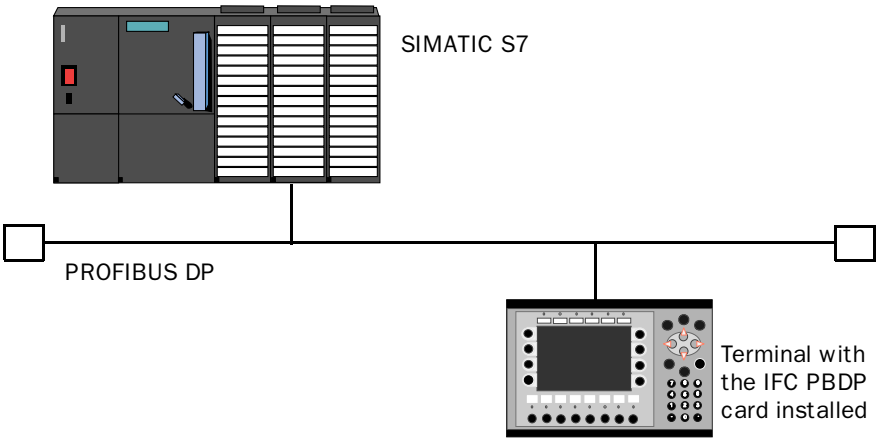
The following parameters are defined for the function block 110:

Parameter	Description
CON_LEN	State the length of the request container and response container in bytes. Must be the same as the setting in the terminal.
FB_IN_OFFSET	State the address to the first byte in the response container in the profibus area.
FB_OUT_OFFSET	State the address to the first byte in the request container in the profibus area.
DB_ADDRESS	State the number of the data block used.

For more information, refer to the SIMATIC S7 manual.

Exempel

In this example we use Siemens PLC system S7 and the PLC program on the type diskette. The example describes in which order you make the settings, and connections to get the correct communication.



1. Install the terminal according to the Installation manual delivered with the terminal.
2. Configure the terminal with the software package MAC Programmer+. The settings for the IFC PBDP card is made in the **Setup** menu under **Expansion slots**.
3. Start the SC (system configuration) in STEP7.
4. Configure the master, baudrate, station number, number of bytes in the transfer container, etc. For more information, refer to the software manual. Type files for the terminal are available on the IFC PBDP diskette.
5. Load the configuration to the S7. See the SIMATIC S7 manual.
6. Load the enclosed PLC program to the SIMATIC S7.
7. Connect the cable between the SIMATIC S7 system and the IFC PBDP card in the terminal.

8. Put the PLC system and the MMI terminal in run mode.

Note!
If you try to open a non-existent data block the PLC system will stop. For more information, refer to the SIMATIC manual.

Printout of PLC program section

In the chapter *Appendix for printouts* at the back of the manual you will find a printout of the PROFIBUS DP function block.

7 The MMI profile

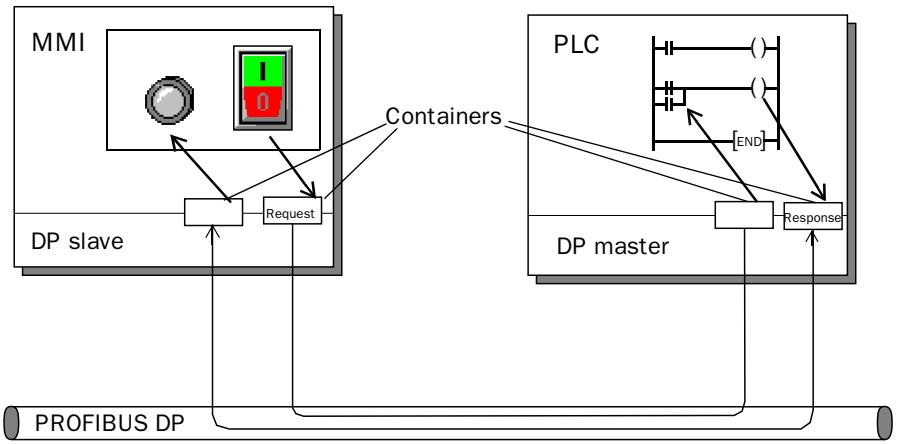
This chapter describes setup of the MMI profile, and is for the benefit of readers who want to learn more about data exchange via the MMI profile.

The MMI profile allows exchange of an unlimited amout of data, and also allows the terminal to access all type of devices in the PLC system.

Together with the card a type diskette is supplied containing PLC program for communication with the Mitsubishi MELSEC A system via the AJ71PB92 module and Siemens PLC system SIMATIC S5 and S7.

PROFIBUS-DP allows a maximum byte length of 200 bytes in and 200 out per station. The MMI profile uses an input area and an output area. These areas are hereafter referred to as containers. The MMI uses the container to access the PLC.

For more detailed information on the MMI profile see the specifications from the Profibus Organisation.



7.1 The data exchange

- The MMI terminal is a slave in the PROFIBUS-DP net.
- The PLC system is the master.
- The MMI terminal requests data from the PLC system through the input container.
- The PLC program serves the MMI terminal with data through the output container.
- Handshaking between the MMI terminal and the PLC system is performed through a Control byte in the containers.
- The MMI terminal can access all types of PLC devices.

When the MMI terminal toggles the control byte, the PLC knows that the MMI terminal wants to exchange data.

7.2 The request and response containers

The container starts at address 0 with the control byte. The control byte is used for handshaking and for communication failure detection. Addresses 1-3 are reserved for Fast bytes. These are not used in the MMI terminal.

Addresses 4 to 200 are used for communication. The MMI terminal put indexes here (3 byte each) that refer to the PLC devices that the MMI wants to read or write. The PLC system, on the other hand, will put the data here from the PLC devices that the MMI terminal has asked for. If the MMI terminal wants to write to a PLC device, the data is stored immediately after the index.

Request container		Response container	
00	Control byte	**00**	Control byte
01	Not used	**01**	Not used
02	Not used	**02**	Not used
03	Not used	**03**	Not used
04	Index 1 Read	**04**	Data for index 1
05	- -	**05**	- -
06	- -	**06**	Data for index 2-
07	Index 2 Read	**07**	- -
08	- -	**08**	- -
09	- -	**09**	- -
10	Index 3 Write	**10**	- -
11	- -	**11**	- -
12	- -	**12**	- -
13	Data byte for index 3	**13**	Free
14	Data byte for index 3	**14**	Free
....200		**....200**	Free

The control byte in the request container

The request container contains a message from the MMI terminal to the PLC system.

7	6	5	4	3	2	1	0
Request	COM	Toggle	Error	Acknowledge bits, not used			

Request

The request byte is used for handshaking between the units. The bit toggles when the MMI terminal wants information for the PLC system.

COM

The COM bit is set by the MMI terminal. If communication breaks the bit will be reset.

Toggle

The toggle bit is always set to the opposite value as the toggle bit in the request container.

Error

This bit is not used.

Acknowledge

These bits are not used.

The control byte in the response container

The response container contains the response from the PLC system to the MMI terminal.

7	6	5	4	3	2	1	0
Response	COM	Toggle	Error	Acknowledge bits, not used			

Response

Is set to the same value as request when data is ready for transfer to the MMI terminal.

COM

The OM bit is set by the PLC program. If communication breaks the bit will be reset.

Toggle

The toggle bit is always set to the same value as the toggle bit in the request container.

Error

This bit is not used.

Action

These bits are not used.

7.3 The index structure

The index is built up of 3 bytes. The index contains 4 parts of information:

– If the device should be read or written.

– Which type of device (input, data register, timer etc.)

– Number of device (e.g. input 5).

– Data length (from one bit up to 16 bytes).

7	6	5	4	3	2	1	0
Write	Ln2	Ln1	Ln0	PLC device type			
Index number bit 15-8							
Index number bit 7-0							

PLC device type

States the type of PLC device according to the following table.

Number	MELSEC A		SIMATIC S5		SIMATIC S7	
	Data type MELSEC	Data type IEC	Data type English	Data type German	Data type IEC	Data type SIMATIC
1	M	%MX0.	F	M	M	M
2	X	%IX	I	E	I	E
3	Y	%QX	Q	A	Q	A
4	B	%MX1.				
8	D	%MW0.	DB	DB	DB	DB
9	R	%MW2.				
10	T	%MW3.				
11	C	%MW4.				
12	W	%MW1.				

Ln0-Ln2

States the data length according to the following table.

Ln2	Ln1	Ln0	Length
0	0	0	bit
0	0	1	1 byte
0	1	0	2 bytes
0	1	1	4 bytes
1	0	0	6 bytes
1	0	1	8 bytes
1	1	0	12 bytes
1	1	1	16 bytes

Sequence of events

– The MMI terminal decides which variables are to be read/written.

– The terminal toggles the request flag in the control byte.

– In the next PROFIBUS cycle, the PLC notices that the request flag has been changed.

– For each read index, the values of the requested devices are copied to the response container.

– Then the response flag in the response container is set to the same value as the request flag in the request container.

– In the next PROFIBUS cycle, the MMI terminal notices that the request flag and the response flag are the same which means that there is data for the terminal.

– The received values will now be used by the objects in the terminal.

8 Appendix

8.1 The type diskette

The files on the type diskette:

IFCPBDTE.200	Card specification file for DOS software ET200 and STEP 7, English.
IFCPBDTD.200	Card specification file for DOS software ET200 and STEP 7, German.
IFC-PBDP.GSD	Card specification file for Windows software COM ET200.
S5	Directory containing the SIMATIC S5 project.
S7	Directory containing the SIMATIC S7 project.

8.2 SIMATIC S5 project

To change this, simply adjust the PLC program. The following devices are used by the program.

MB200-203, MB206-217, MB220-225, MB230-233, MB240-241 and T2.

8.3 SIMATIC S7 project

To change this, simply adjust the PLC program. The following devices are used by the program.

MB100-101, MB110, MB200-217, MB220-225, MB230-233, MB240-247 and T2.

Index

SVENSK

Manual PROFIBUS DP

Förord

Denna manual är en installations och funktionsbeskrivning för PROFIBUS DP kommunikationsmodulen IFC PBDP.
Modulen kan användas till E300, E700 och E710 terminalerna.
Förutom denna manual finns också följande manualer tillgängliga.

- E700, Installation
- E710, Installation
- E700/E710, Manual

För information om Siemens PLC-system hänvisas till respektive manual.

Säkerhetsföreskrifter

Allmänt

– Kontrollera de levererade produkterna för att upptäcka eventuella transportskador. Meddela din leverantör om skador upptäcks.
– Produkten uppfyller kraven enligt artikel fyra i EMC-direktivet 89/336/EEC.
– Produkten får ej användas i explosiv miljö.
– All modifiering, ändring och ombyggnad av produkten är förbjuden.
– Endast reservdelar godkända av G & L Beijer Electronics AB får användas.
– Läs användarbeskrivningen noga innan produkten används.
– Utrustningen måste hanteras av personal med adekvat utbildning.

Vid installation

– Produkten är konstruerad för fasta installationer.
– Installera produkten enligt medföljande installationsbeskrivning.
– Jordning skall ske enligt medföljande installationsbeskrivning.
– Installation skall göras av personal med adekvat utbildning.
– Högspännings-, signal-, och spänningskablar måste separeras.
– Produkten bör ej monteras i direkt solljus.

Vid användning

– Håll utrustningen ren.
– Nödstoppsfunktioner eller andra säkerhetsfunktioner får ej styras från MAC-terminalen.
– Tangenter, displayglas etc. får ej påverkas med vassa föremål.

Service och underhåll

– Garanti gäller enligt avtal.
– Använd lätt rengöringsmedel och mjuk trasa för att rengöra displayglaset och fronten.
– Använd batterier enligt specifikation från G & L Beijer Electronics AB. Batteribyte ska utföras av personal med adekvat utbildning. Person som utför arbetet måste vara jordad under tiden utbytet sker, t ex med jordat handledsband.
– Reparationer ska utföras av auktoriserad personal.

Vid nedmontering och skrotning

– Återvinning av produkten eller delar av produkten skall ske enligt gällande regler i respektive land.
– Beakta att följande komponenter innehåller farliga ämnen: elektrolytkondensatorer samt display.

Innehåll

1 Introduktion

PROFIBUS DP är en leverantörsoberoende, öppen industrifältbuss som kan användas i många olika applikationer. Det är en etablerad teknologi med ett stort antal installationer.

Extern fältutrustning installerad för processautomatisering som sensorer, ställdon, sändare, drivenheter och PLC-system använder mer och mer mikroelectronik.

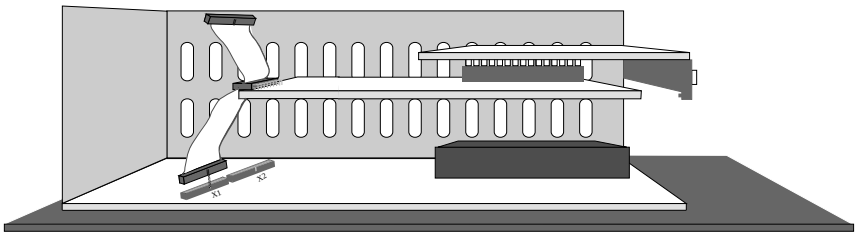
PROFIBUS DP medger att enheter från olika leverantörer kan kommunicera på ett effektivt sätt i ett nätverk. PROFIBUS är standardiserad som Europa Standard pr EN 50170.

Tillsammans med PROFIBUS DP kortet levereras en typdiskett som innehåller bl a PLC-program för kommunikation med Mitsubishi Electric PLC-system MELSEC A samt Siemens PLC-system SIMATIC S5 och SIMATIC S7.

2 Installation

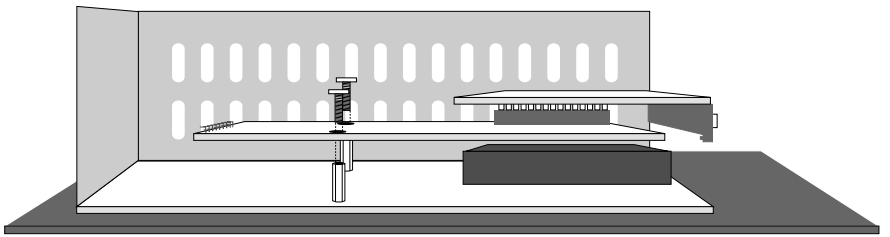
Detta kapitlet beskriver hur IFC PBDP kortet ansluts till MMI-terminalerna. IFC PBDP paketet innehåller IFC PBDP kortet och en typdiskett.

2.1 Anslutning av flatkabeln



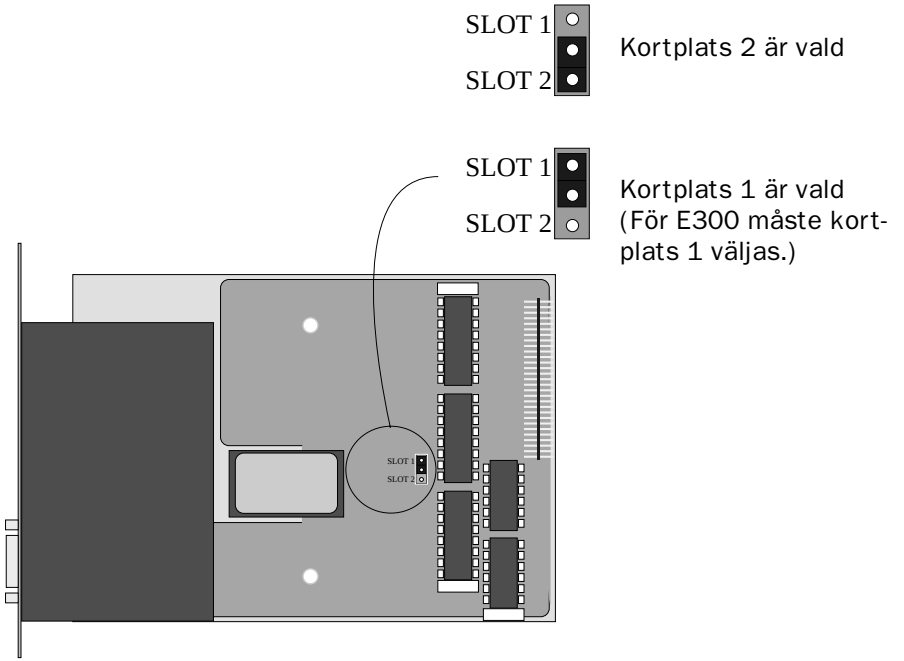
1. Skruva av täckplåten på terminalens baksida samt kortplatspanelen.
2. Anslut flatkabeln till kontakten X1.

2.2 Anslutning av IFC PBDP kortet

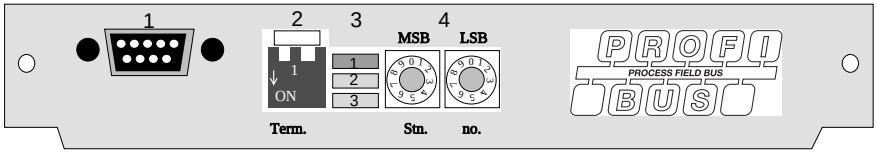


Montera IFC PBDP kortet med medföljande distanser och skruvar.

2.3 Val av fysisk port

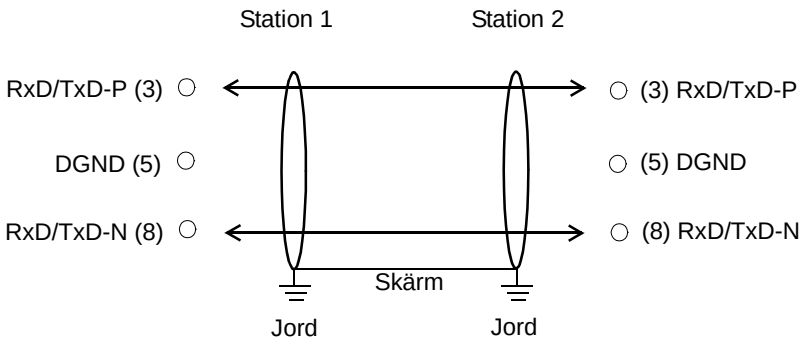


2.4 Kommunikationsinställningar för IFC PBDP kortet



1. Kontakt för anslutning av kommunikationskabel.
2. Bussterminering. Sätts i läge ON på den första och sista enheten i nätverket. Den första enheten i nätverket är oftast masterenheten i PLC-systemet.
3. 1: Röd, **ERR**, Konfigurerings- eller kommunikationsfel. Lysdioden är röd tills enheten är konfigurerad. Indikerar time out.
   2: Grön, **PWR**, Spänningsmatning, 5 VDC OK.
   3: Grön, **DIA**, Diagnostikfel. Används inte.
4. Anger stationsnummer.

2.5 Kabel till PROFIBUS DP



2.6 Tekniska data

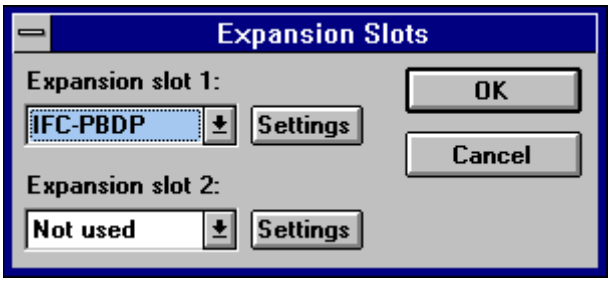
I/O area storlek	32-200 byte
Överföringshastighet	9600 bit/s - 12 MBit/s
Identitetskod	1002
Max. antal noder utan repeater	32
Max. antal noder med repeater	96
Max. kabellängd (med repeater)	3000m, 9.6 kb
Max. kabellängd (utan repeater)	200m, 12 Mb

Kabeln Unitronic-Bus L2/F.I.P är testad och har följande prestanda:

Kapacitans	30 nF/km
Impedans	150 Ohm (3-20 MHz)
Resistans	115 Ohm/km

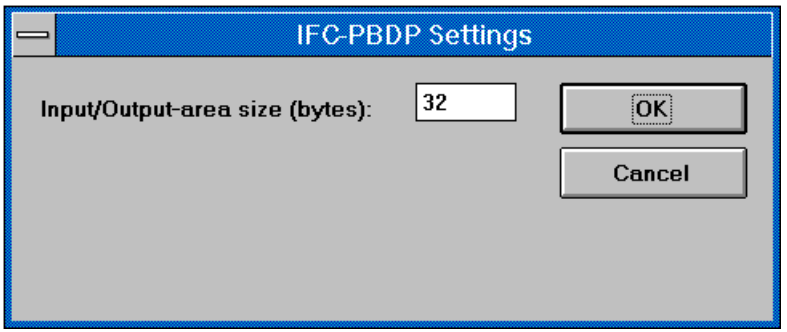
3 Konfigurering av MMI-terminalen

Terminalen konfigureras med *MAC Programmer*+. Inställningarna görs under **Setup** menyn, **Expansion slots**.



Expansion slot 1-2:
Välj IFC PBDP för aktuell kortplats. Inställningen måste vara samma som inställningen av bygeln på IFC PBDP kortet.

Settings:
Inställningar för MMI profilen.



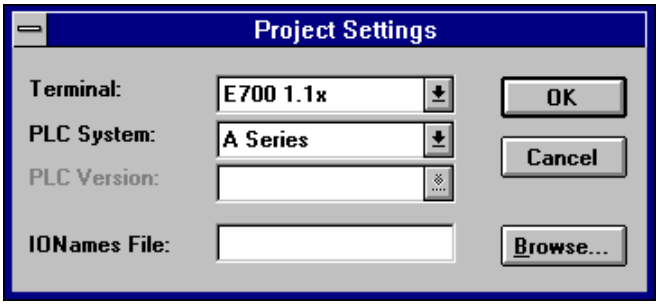
Input/Output-area size (bytes):
Anger storleken på ingångs- och utgångsarean i antal bytes. Grundinställningen är 32 bytes.

4 Anslutning till MELSEC A

Med PROFIBUS DP kortet kan terminalen kommunicera med Mitsubishi PLC-system i A-serien via modulen A(1S)J71PB92. För information om A-seriens PLC-system och terminalerna hänvisas till respektive manual.

4.1 Val av PLC-system

I dialogen Project Settings i MAC Programmer+ väljer du vilket PLC-system MMI-terminalen ska vara uppkopplad mot. Välj A Series.



4.2 I/O hantering

Terminalen kan hantera följande datatyper i MELSEC A.

Datatyper MELSEC	Datatyper IEC	Betydelse
X	**%IX**	Ingång
Y	**%QX**	Utgång
M	**%MX**	Minnescell
B	**%MX**	Länkminnescell (MELSEC NET)
D	**%MW**	Dataregister
W	**%MW**	Länkregister (MELSEC NET)
R	**%MW**	Filregister
T	**%MW**	Tidkrets, aktuellt värde
C	**%MW**	Räknare, aktuellt värde

Observera!
MMI-profilen använder ett antal minnesceller och dataregister för intern hantering. Dessa kan inte användas till någonting annat i PLC-programmet.

Digitala signaler

Datatyp MELSEC	Datatyp IEC
Xn	**%IX**b
Yn	**%QX**b
Mn	**%MX0.**b
Bn	**%MX1.**b

n=adress, b=bitnummer.

Analoga signaler

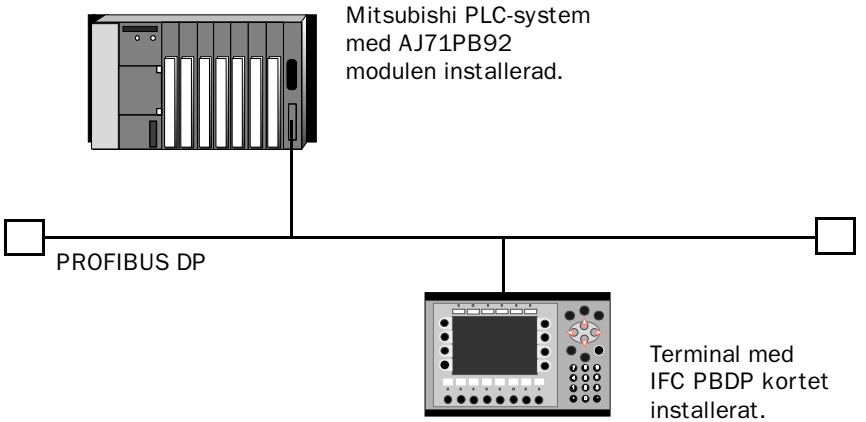
Datatyp MELSEC	Datatyp IEC
Dn	**%MW0.**n
Wn	**%MW1.**n
Rn	**%MW2.**n
Tn	**%MW3.**n
Cn	**%MW4.**n

n=adress

För mer information om adressering till Mitsubishi Electric PLC-system MELSEC A hänvisas till respektive manual.

4.3 Exempel

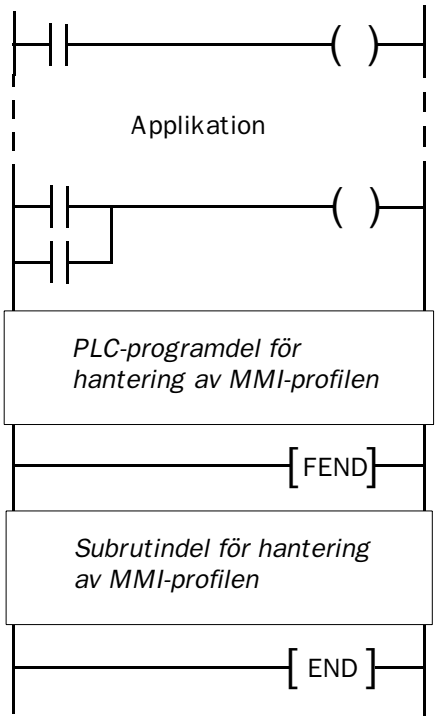
I detta exemplet använder vi Mitsubishi modulen AJ71PB92 och PLC-programmet som finns på medföljande typdiskett. Exemplet beskriver i vilken ordning du gör inställningarna och anslutningarna för att få en korrekt kommunikation.



1. Installera terminalen enligt Installationsmanualen som levereras med terminalen.
2. Konfigurera terminalen via programpaketet MAC Programmer+. Inställningarna för IFC PBDP kortet görs i **Setup** menyn under **Expansion slots**. Se kapitlet *Konfigurering av MMI-terminalen*.
3. Starta PROFIBUS konfigureringsprogramvara (exempelvis COM ET200).
4. Konfigurera mastern, överföringshastighet, stationsnummer, antal bytes i överföringsarean etc. För mer information hänvisas till manualen för programvaran. Typfilen för terminalen finns på IFC PBDP disketten.
5. Skapa en binärfil. För information hänvisas till manualen för programvaran COM ET200 för DOS.

6. Starta programmet SWOIX-DPLD från DOS. Skicka ner binärfilen till AJ71PB92 modulen via SWOIX-DPLD programmet. För mer information hänvisas till manualen för programmet SWOIX-DPLD.
7. Skicka ner medföljande PLC-program till A-seriens CPU via MELSEC MEDOC.
8. Anslut kabeln mellan AJ71 PB 92 modulen i PLC-systemet och IFC PBDP kortet i terminalen.
9. Sätt PLC-systemet och MMI-terminalen i driftläge.

Nedanstående figur visar hur programdel och subrutindel placeras i PLC-programmet.



Utskrift av PLC-programdel

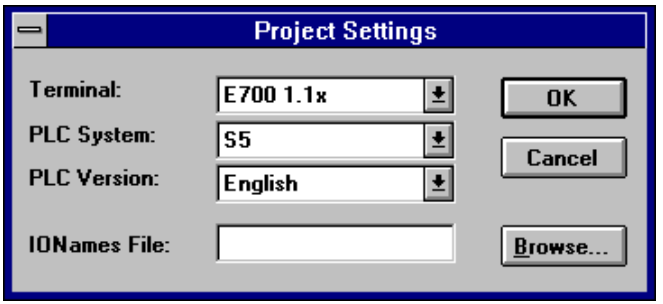
I kapitlet *Appendix for printouts* längst bak i manualen finns en utskrift av PLC-programdelen PROFILE för PROFIBUS DP kommunikation.

5 Anslutning till SIMATIC S5

Med PROFIBUS DP kortet kan MMI-terminalen kommunicera med Siemens PLC-system SIMATIC S5. Detta kapitlet beskriver hur funktionsblocken är uppbyggda, vilka I/O terminalen kan adressera samt hur PLC-programmet anropas. För mer information om SIMATIC S5 hänvisas till Siemens dokumentation för systemet SIMATIC S5.

5.1 Val av PLC-system

I dialogen Project Settings i MAC Programmer+ väljer du vilket PLC-system MMI-terminalen ska vara uppkopplad mot. Välj S5.



5.2 I/O hantering

Terminalen kan hantera följande datatyper i SIMATIC S5:

Datatyp Engelska	Datayp Tyska	Betydelse
F	**M**	Flagga
Q	**A**	Utgång
I	**E**	Ingång
DB	**DB**	Datablock

DB (Datablock) i SIMATIC S5 kan vara max 256 ord långa. Terminalen kan adressera alla DB i PLC-systemet.

Alla datatyper består av byte-areor. Adresseringen sker alltid med avseende på bytes oavsett om det är 1, 16 eller 32 bitar. Adresserna är alltid decimala, 0-65535.

För information om instruktioner i SIMATIC S5 hänvisas till SIMATIC manualen.

Digitala signaler

För digitala signaler anges vilken bit i byten som avses. T ex betyder I50.3 bit 3 i ingångsbyte 50.

Datatyp Engelsk	Datatyp Tysk
Ixxxxx.b	**E**xxxxx.b
Qxxxxx.b	**A**xxxxx.b
Fxxxxx.b	**M**xxxxx.b

xxxxx=adress 0-65535, b=bitnummer 0-7

Analoga signaler

För 16-bitars tal anges suffixet W. T ex betyder MW100 att 2 bytes från minnesbyte 100-101 tas.

Datatyp Engelsk	Datatyp Tysk
IWxxxxx	**EW**xxxxx
QWxxxxx	**AW**xxxxx
FWxxxxx	**MW**xxxxx
DBno.**DW**adr	**DB**no.**DW**adr

xxxxx=adress 0-65535, no=databasnummer 0-255 och adr=dataord inom databasen 0-255.

5.3 Beskrivning av PLC-programdelen

PLC-programdelen består av tre funktionsblock plus ett block (OB1) som anropar funktionsblock 190.

Programblock

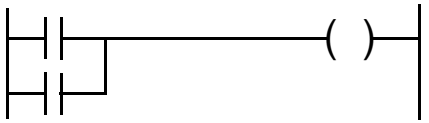
PLC-programdelen som finns på typdisketten innehåller tre funktionsblock och ett huvudprogram:

Funktionsblock	Förklaring
OB1	Huvudprogram. Anropar funktionsblock 190.
FB 190	Detta block anropas av OB1 och sköter hanteringen av MMI-profilen.
FB 191	Detta block läser 1 index.
FB 192	Detta block skriver 1 index.

Nedanstående figur visar hur OB1 placeras i PLC-programmet.

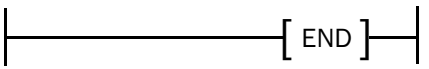
OB1
Segment 1

Segment 3



Segment 4

```
0000      :JU    FB 190
0001      :PROFILE
0002 Name :      KF +32
0003 LEN  :      KF +6
0004 WRI  :      KF +6
0005 INT  :      FY 100
0006 HERR :      FY 101
0007 TEMS :      T1
```



Huvudprogrammet, OB1

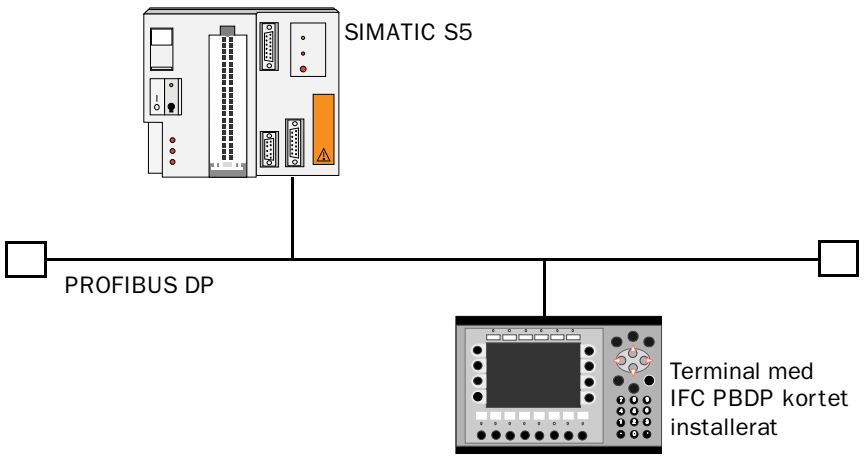
OB1 är huvudprogrammet där parametrar definieras för anrop av övriga funktionsblock. Följande parametrar definieras för funktionsblocket 190:

Parameter	Förklaring
LEN	Anger längden på areorna för begäran och svar i bytes. Måste vara samma som inställningen i terminalen.
READ	Anger adressen till första byten i arean för svar i PROFIBUS-arean.
WRI	Anger adressen till första byten i arean för begäran i PROFIBUS-arean.
INT	Intern byte
HERR	Anger det register som ska innehålla eventuell felkod från FB 190.
TMS	Time out i sekunder. Bryts kommunikationen med terminalen under längre tid än angivet värde i parametern TMS ges felkod i parametern HERR.

För mer information hänvisas till Siemens manual för SIMATIC S5.

Exempel

I detta exemplet använder vi Siemens PLC-system SIMATIC S5 och PLC-programmet som finns på typdisketten. Exemplet beskriver i vilken ord-ning du gör inställningarna och anslutningarna för att få rätt kommunikation.



1. Installera terminalen enligt Installationsmanualen som levereras med terminalen.
2. Konfigurera terminalen via programpaketet MAC Programmer+. Inställningarna för IFC PBDP kortet görs i **Setup** menyn under **Expansion slots**.
3. Starta COM ET200 konfigureringsprogramvara.
4. Konfigurera mastern, överföringshastighet, stationsnummer, antal bytes i överföringsarean etc. För mer information hänvisas till manualen för programvaran. Typfiler för terminalen finns på IFC PBDP disketten
5. Skicka ner konfigurationen till S5. Se Siemens manual för S5.
6. Skicka ner medföljande PLC-programdel till S5.

7. Anslut kabeln mellan S5 systemet och IFC PBDP kortet i terminalen.
8. Sätt PLC-systemet och MMI-terminalen i driftläge.

Observera!
Försöker du öppna ett datablock som inte finns stannar PLC-systemet. För mer information hänvisas till SIMATIC manualen.

Utskrift av PLC-programdelen

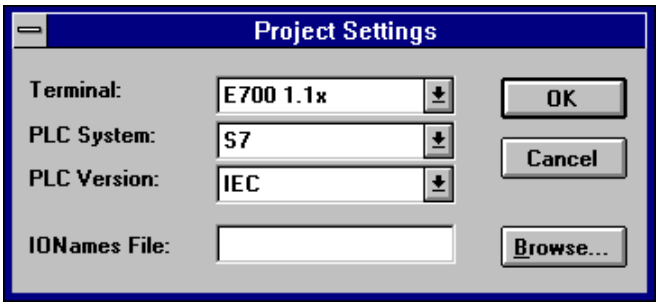
I kapitlet *Appendix for printouts* längst bak i manualen finns en utskrift av PROFIBUS funktionsblocken.

6 Anslutning till SIMATIC S7

Med PROFIBUS DP kortet kan MMI-terminalen kommunicera med Siemens PLC-system SIMATIC S7. Detta kapitlet beskriver hur funktionsblocken är uppbyggda, vilka I/O terminalen kan adressera samt hur PLC-programmet anropas. För mer information om S7 hänvisas till Siemens dokumentation för SIMATIC S7.

6.1 Val av PLC-system

Idialogen Project Settings i MAC Programmer+ väljer du vilket PLC-system MMI-terminalen ska vara uppkopplad mot. Välj S7.



6.2 I/O hantering

Terminalen kan hantera följande datatyper i SIMATIC S7:

Datatyp IEC	Datayp SIMATIC	Betydelse
M	**M**	Flagga
Q	**A**	Utgång
I	**E**	Ingång
DB	**DB**	Datablock

DB (Datablock) i S7 kan vara hur stora som helst. Endast projektminnet begränsar storleken. Terminalen kan adressera ett DB i PLC-systemet.

Alla datatyper består av byte-areor. Adresseringen sker alltid med avseende på bytes oavsett om det är 1, 16 eller 32 bitar. Adresserna är alltid decimala, 0-65535.

För information om instruktioner i SIMATIC S7 hänvisas till SIMATIC manualen.

Digitala signaler

För digitala signaler anges vilken bit i byten som avses. T ex betyder I50.3 bit 3 i ingångsbyte 50.

Datatyp IEC	Datatyp SIMATIC
Ixxxxx.b	**E**xxxxx.b
Qxxxxx.b	**A**xxxxx.b
Mxxxxx.b	**M**xxxxx.b

xxxxx=adress 65535, b=bitnummer 0-7

Analoga signaler

För 16-bitars tal anges suffixet W. T ex betyder MW100 att 2 bytes från minnesbyte 100-101 tas.

Datatyp IEC	Datatyp SIMATIC
IWxxxxx	**EW**xxxxx
QWxxxxx	**AW**xxxxx
MWxxxxx	**MW**xxxxx
DBWxxxxx	**DBW**xxxxx

xxxxx=adress 0-65535

6.3 Beskrivning av PLC-programdelen

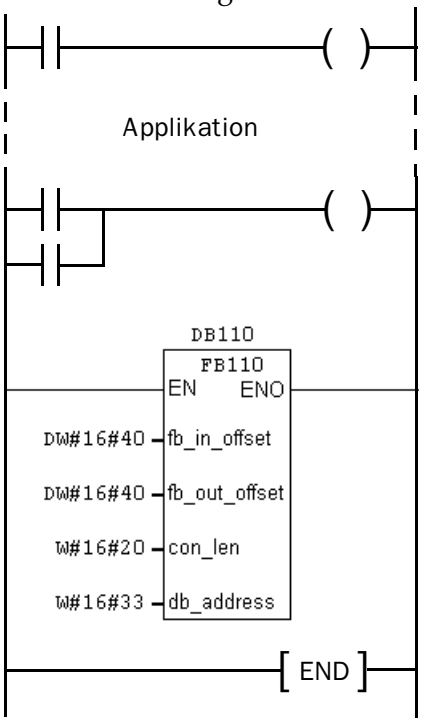
PLC-programdelen består av ett funktionsblock plus ett huvudprogram (OB1) som anropar funktionsblocket.

Programblock

PLC-programdelen som finns på typdisketten innehåller tre funktionsblock och ett huvudprogram:

Funktionsblock	Förklaring
OB1	Huvudprogram. Anropar funktionsblock 110.
FB 110	Detta block anropas av OB1 och sköter hanteringen av MMI-profilen.
FC 111	Detta block läser 1 index.
FC 112	Detta block skriver 1 index.
DB 51	Datablock som används för analoga signaler.

Nedanstående figur visar hur OB1 placeras i PLC-programmet.



Huvudprogrammet, OB1

OB1 är huvudprogrammet där parametrar definieras för anrop funktionsblock 110.

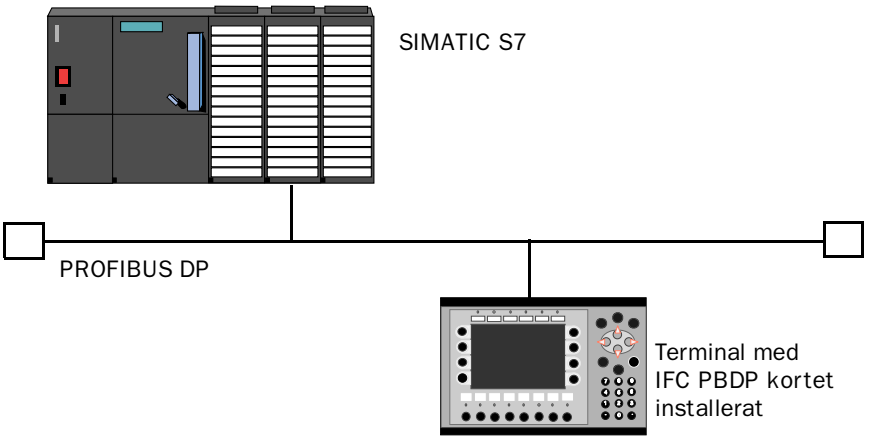
Följande parametrar definieras för funktionsblock 110:

Parameter	Förklaring
CON_LEN	Anger längden på areorna för begäran och svar i bytes. Måste vara samma som inställningen i terminalen.
FB_IN_OFFSET	Anger adressen till första byten i arean för svar i PROFIBUS-arean.
FB_OUT_OFFSET	Anger adressen till första byten i arean för begäran i PROFIBUS-arean.
DB_ADDRESS	Anger numret på datablocket som används.

För mer information om parametrarna hänvisas till Siemens manual för SIMATIC S7.

Exempel

I exemplet använder vi Siemens PLC-system SIMATIC S7 och PLC-programdelen som finns på typdisketten. Exemplet beskriver i vilken ordning du gör inställningarna och anslutningarna för att få rätt kommunikation.



1. Installera terminalen enligt Installationsmanualen som levereras med terminalen.
2. Konfigurera terminalen via programpaketet MAC Programmer+. Inställningarna för IFC PBDP kortet görs i **Setup** menyn under **Expansion slots**.
3. Starta SC (system configuration) i STEP7.
4. Konfigurera mastern, överföringshastighet, stationsnummer, antal bytes i överföringsarean etc. För mer information hänvisas till manualen för programvaran. Typfiler för terminalen finns på IFC PBDP disketten.
5. Skicka ner konfigureringen till S7. Se Siemens manual för S7.
6. Skicka ner medföljande PLC-programdel till S7.

7. Anslut kabeln mellan S7 systemet och IFCPBDP kortet i terminalen.
8. Sätt PLC-systemet och MMI-terminalen i driftläge.

Observera!
Försöker du öppna ett datablock som inte finns stannar PLC-systemet. För mer information hänvisas till SIMATIC manualen.

Utskrift av PLC-programdelen

I kapitlet *Appendix for printouts* längst bak i manualen finns en utskrift av PLC-programdelen för PROFIBUS DP kommunikation.

7 MMI-profilen

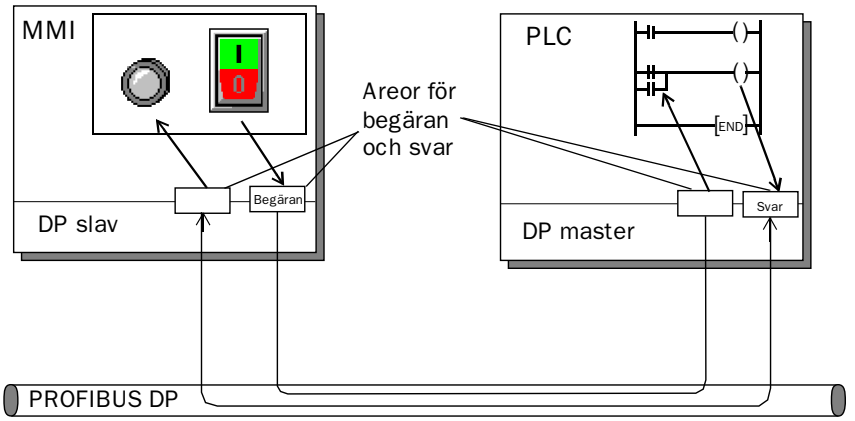
Detta kapitlet beskriver hur MMI-profilen är uppbyggd och är riktad till de användare som vill veta lite mer om datautbyte via MMI-profilen.

MMI-profilen tillåter utbyte av obegränsat antal data. Dessutom tillåter den terminalen att accessa alla datatyper i PLC-systemet.

Tillsammans med kortet levereras en typdiskett som innehåller PLC-programdelar för kommunikation med Mitsubishi MELSEC A system via AJ71PB92 modulen samt Siemens PLC-system SIMATIC S5 och S7.

PROFIBUS-DP tillåter max byte längd på 200 bytes in och 200 bytes ut per station. MMI-profilen använder en area för begäran och en area för svar. Areorna används för att accessa PLC-systemet.

För mer information om MMI-profilen hänvisas till specifikationer från the Profibus Organisation.



7.1 Datautbyte

– MMI-terminalen är alltid slav i ett PROFIBUS DP nät.

– PLC-systemet är master.

– MMI-terminalen begär data från PLC-systemet via arean för begäran.

– PLC-program förser MMI-terminalen med data via arean för svar.

– Handskakning mellan MMI-terminalen och PLC-systemet sköts via en kontrollbyte i respektive area.

– MMI-terminalen kan accessa alla datatyper.

När MMI-terminalen växlar status på kontrollbyten vet PLC-systemet att MMI-terminalen vill utbyta data.

7.2 Areorna för begäran och svar

MMI-profilen är uppbyggd av areor mellan vilka datautbytet sker.

Areorna startar på adress 0 med en kontrollbyte. Kontrollbyten används för handskakning och för detektering av kommunikationsfel. Adresserna 1-3 är reserverade för Snabba bytes. Dessa används inte i terminalen.

Adresserna 4 till 200 används för kommunikation. Här sätter MMI-terminalen index (3 byte/index) i arean för begäran, som refererar till de PLC-adresser som MMI-terminalen vill läsa eller skriva till. PLC-systemet lägger den data MMI-terminalen önskar från PLC-systemet i motsvarande index i arean för svar. Om MMI-terminalen vill skriva till en PLC-adress lagras data direkt efter index i arean för svar.

Area för begäran		**Area för svar**	
00	Kontrollbyte	**00**	Kontrollbyte
01	Används inte	**01**	Används inte
02	Används inte	**02**	Används inte
03	Används inte	**03**	Används inte
04	Index 1 Läs	**04**	Data för index 1
05	- -	**05**	- -
06	- -	**06**	Data för index 2
07	Index 2 Läs	**07**	- -
08	- -	**08**	- -
09	- -	**09**	- -
10	Index 3 Skriv	**10**	- -
11	- -	**11**	- -
12	- -	**12**	- -
13	Databyte för index 3	**13**	Ledig
14	Databyte för index 3	**14**	Ledig
....200		**....200**	Ledig

Kontrollbyten i arean för begäran

Arean för begäran innehåller meddelande från MMI-terminalen till PLC-systemet.

7	6	5	4	3	2	1	0
Request	COM	Toggle	Error	Acknowledge bits, not used			

Request

Request-biten används för handskakning mellan enheterna. Biten växlar status när MMI-terminalen vill ha information från PLC-systemet.

COM

COM-biten sätts av MMI-terminalen. Bryts kommunikationen nollställs COM-biten.

Toggle

Toggle-biten har alltid motsatt status som toggle-biten i arean för begäran.

Error

Denna bit används inte.

Acknowledge

Dessa bitar används inte.

Kontrollbyten i arean för svar

Arean för svar innehåller svaret från PLC-systemet till MMI-terminalen.

7	6	5	4	3	2	1	0
Response	COM	Toggle	Error	Acknowledge bits, not used			

Response

Sätts till samma värde som request-biten när data är klart för överföring till MMI-terminalen.

COM

COM-biten sätts av PLC-programmet. Bryts kommunikationen nollställs biten.

Toggle

Toggle-biten sätts alltid till samma status som toggle-biten i arean för begäran.

Error

Denna bit används inte.

Acknowledge

Dessa bitar används inte.

7.3 Strukturen på index

Ett index byggs upp av tre bytes. Indexet innehåller fyra delar med information:

– Om datatypen ska läsas/skrivas.
– Vilken datatyp (ingång, dataregister, tidkrets etc.)
– Datatypens adress (t ex ingång 5).
– Datalängd (från en bit till 16 bytes).

7	6	5	4	3	2	1	0
Skriv	Ln2	Ln1	Ln0	PLC datatyp			
Index nummer bit 15-8							
Index nummer bit 7-0							

PLC datatyp

Anger PLC datatyp enligt följande tabell:

Nummer	MELSEC A		SIMATIC S5		SIMATIC S7	
	Datatyp MELSEC	Datatyp IEC	Datatyp Engelsk	Datatyp Tysk	Datatyp IEC	Datatyp SIMATIC
1	M	%MX0.	F	M	M	M
2	X	%IX	I	E	I	E
3	Y	%QX	Q	A	Q	A
4	B	%MX1.				
8	D	%MW0.	DB	DB	DB	DB
9	R	%MW2.				
10	T	%MW3.				
11	C	%MW4.				
12	W	%MW1.				

Ln0-Ln2

Anger datalängden enligt följande tabell:

Ln2	Ln1	Ln0	Längd
0	0	0	bit
0	0	1	1 byte
0	1	0	2 bytes
0	1	1	4 bytes
1	0	0	6 bytes
1	0	1	8 bytes
1	1	0	12 bytes
1	1	1	16 bytes

Händelseförlopp

– MMI-terminalen bestämmer vilken variabel som ska läsas/skrivas.

– Terminalen växlar status på request flaggan i kontrollbyten.

– Nästa PROFIBUS cykel upptäcker PLC-systemet att request-flaggan har ändrats.

– För varje läsindex kopieras värdet i den begärda datatyper till arean för svar.

– Därefter sätts response-flaggan i arean för svar till samma värde som request-flaggan i arean för begäran.

– Nästa PROFIBUS cykel upptäcker MMI-terminalen att request-flaggan och response-flaggan har samma värde vilket betyder att det finns data till terminalen.

– De mottagna värdena kommer nu att användas av objekten i terminalen.

8 Appendix

8.1 Typdisketten

Följande filer finns på typdisketten:

IFCPBDTE.200	Kortspecifikationsfil för DOS-programvaran ET200 samt STEP 7 på engelska.
IFCPBDTD.200	Kortspecifikationsfil för DOS-programvaran ET200 samt STEP 7 på tyska.
IFC-PBDP.GSD	Kortspecifikationsfil för Windowsprogramvaran COM ET200.
MELSEC	Bibliotek som innehåller MELSEC MEDOC projekt.
S5	Bibliotek som innehåller SIMATIC S5 projekt.
S7	Bibliotek som innehåller SIMATIC S7 projekt.

8.2 MELSEC MEDOC projekt

Projektet kallas PROFILE. Vi antar följande:

– AJ71PB92 modulen är placerad på kortplats 0 i PLC-systemet.

– AJ71PB92 modulen kommunicerar med IFC PBDP med stationsnummer 3.

Följande datatyper kan accessas från terminalen:

Digital: X, Y, M, B
Analog: D, R, W, T, C

Följande datatyper supportas inte: Special D, special M, digital T, digital C och digital F.

För att ändra detta behöver endast PLC-programmet justeras. Följande datatyper används av programmet.

D100-134, D140-171, D180-211 och M100-M115, M120-M121, M200.

8.3 SIMATIC S5 projekt

För att ändra detta behöver endast PLC-programmet justeras. Följande datatyper används av programmet.

MB200-203, MB206-217, MB220-225, MB230-233, MB240-241 och T2.

8.4 SIMATIC S7 projekt

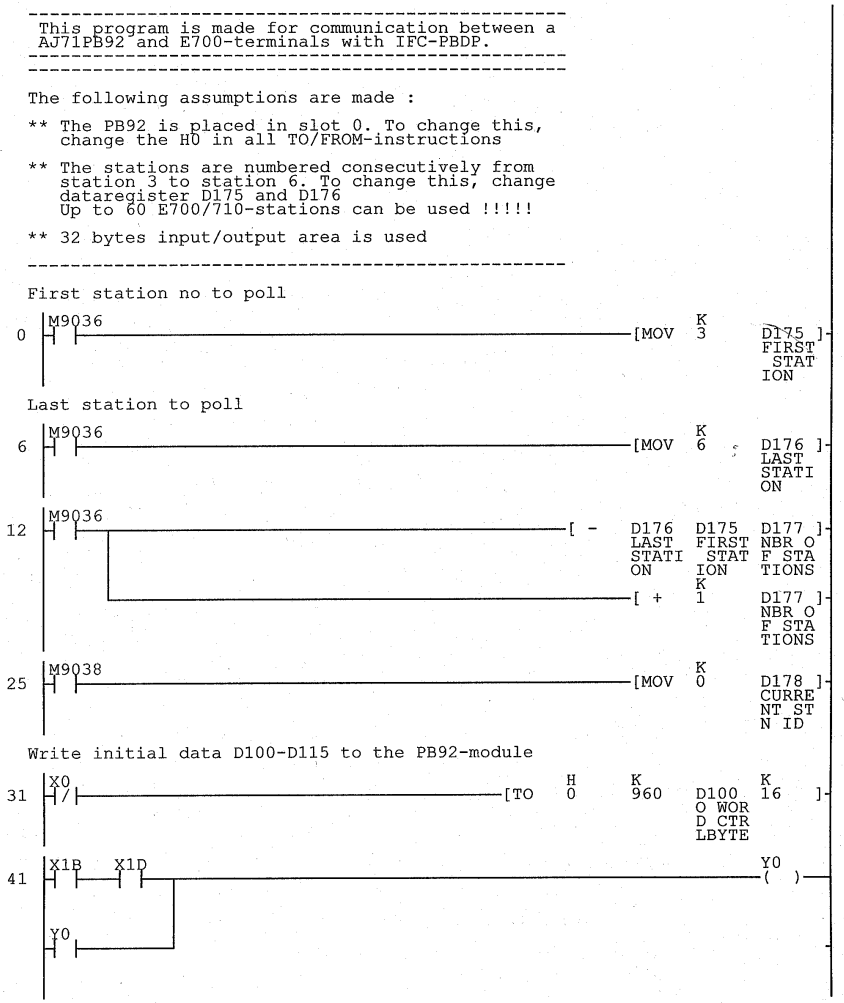
För att ändra detta behöver endast PLC-programmet justeras. Följande datatyper används av programmet.

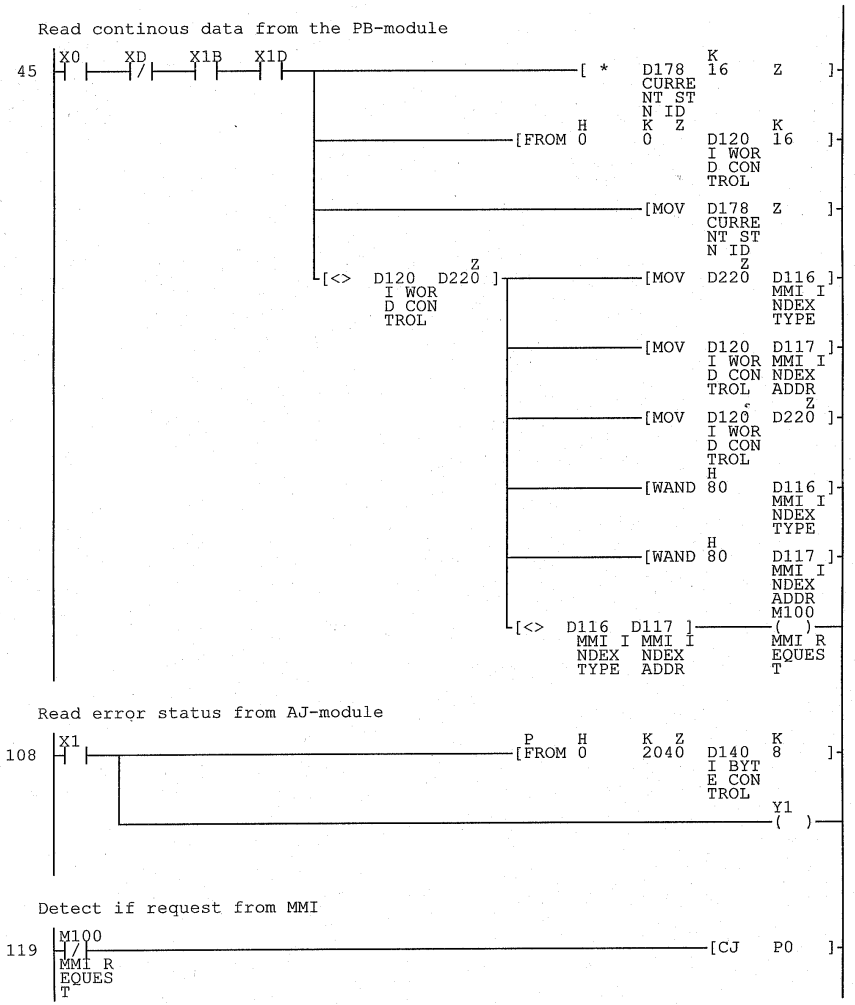
MB100-101, MB110, MB200-217, MB220-225, MB230-233, MB240-247 och T2.

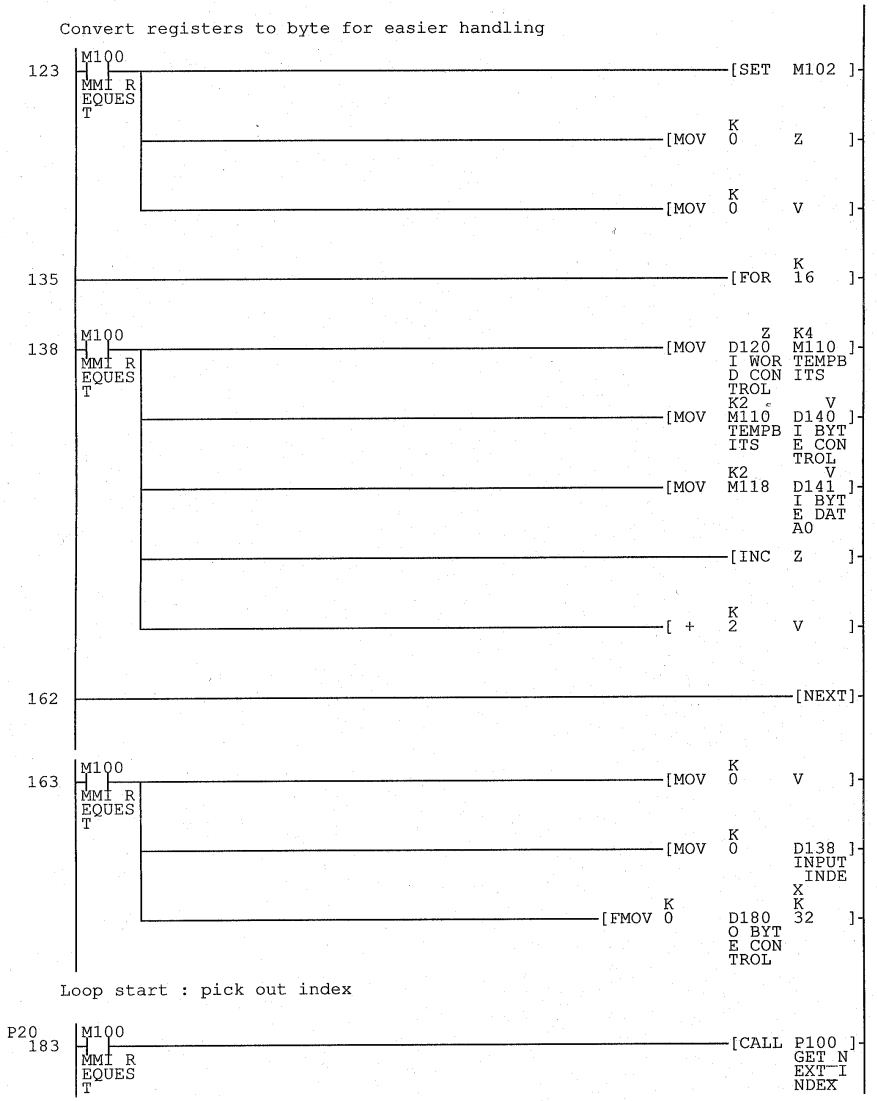
Index

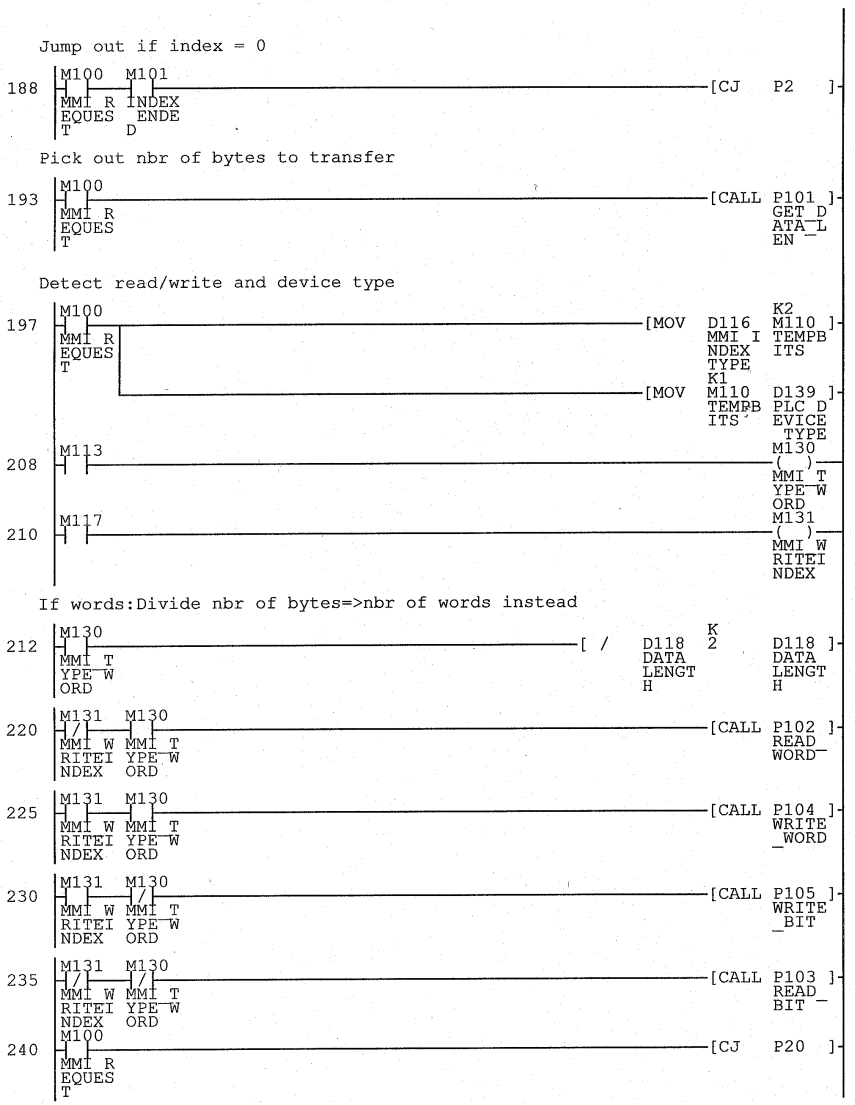
9 Appendix for printouts

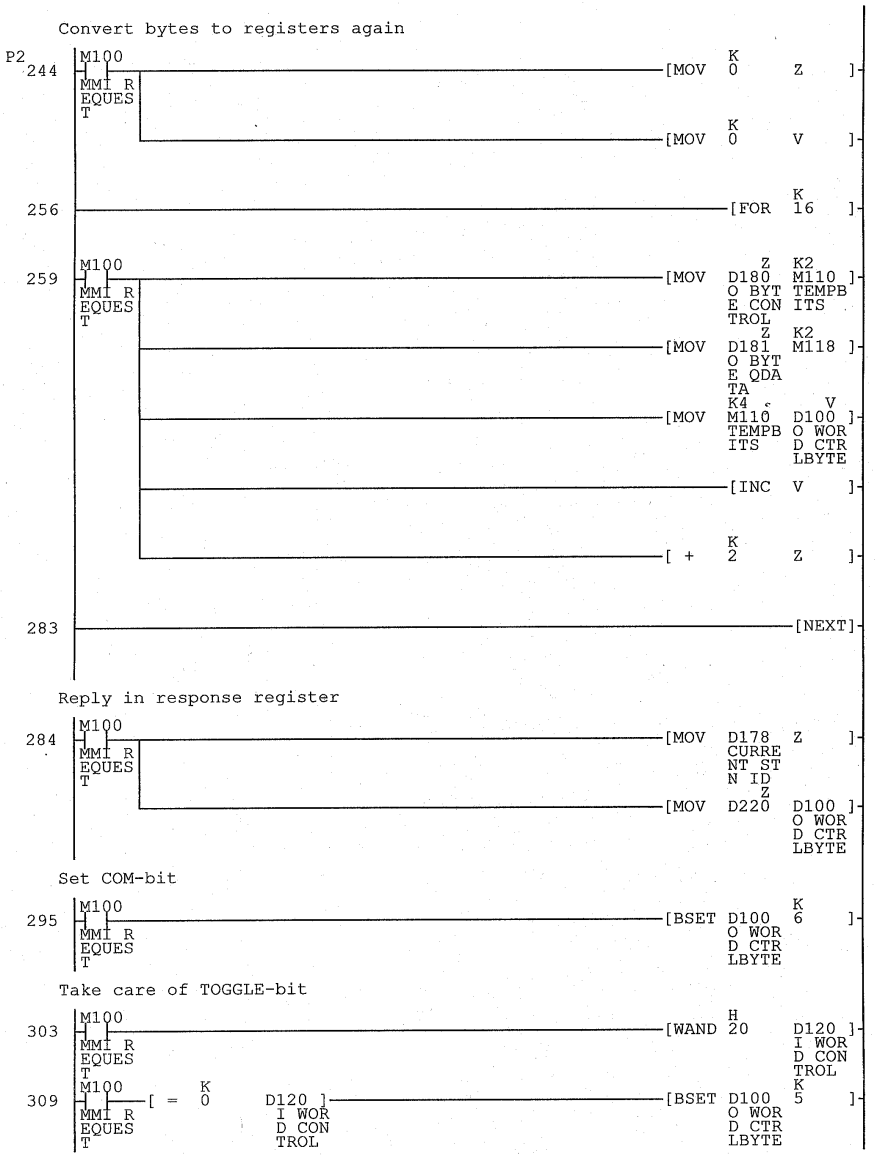
The PLC program section for Mitsubishi MELSEC A

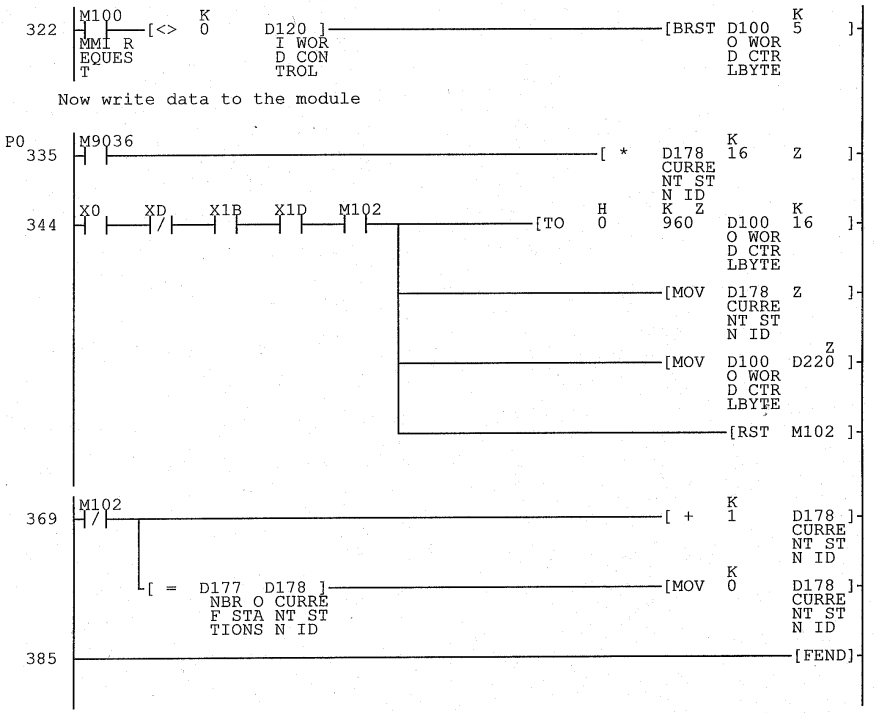



Read continous data from the PB-module
45
X0
XD
X1B
X1D
[* D178 CURRENT STN ID K 16 Z]
[FROM H 0 K Z 0 D120 I WORD CONTROL K 16]
[MOV D178 CURRENT STN ID Z]
[<> D120 I WORD CONTROL Z D220]
[MOV D220 D116 MMI INDEX TYPE]
[MOV D120 I WORD CONTROL D117 MMI INDEX ADDR]
[MOV Z D120 I WORD CONTROL D220]
[WAND H 80 D116 MMI INDEX TYPE]
[WAND H 80 D117 MMI INDEX ADDR]
[<> D116 MMI INDEX TYPE D117 MMI INDEX ADDR]
( ) M100 MMI REQUEST
Read error status from AJ-module
108
X1
[FROM P H 0 K Z 2040 D140 I BYTE CONTROL K 8]
( ) Y1
Detect if request from MMI
119
M100 MMI REQUEST
[CJ P0]

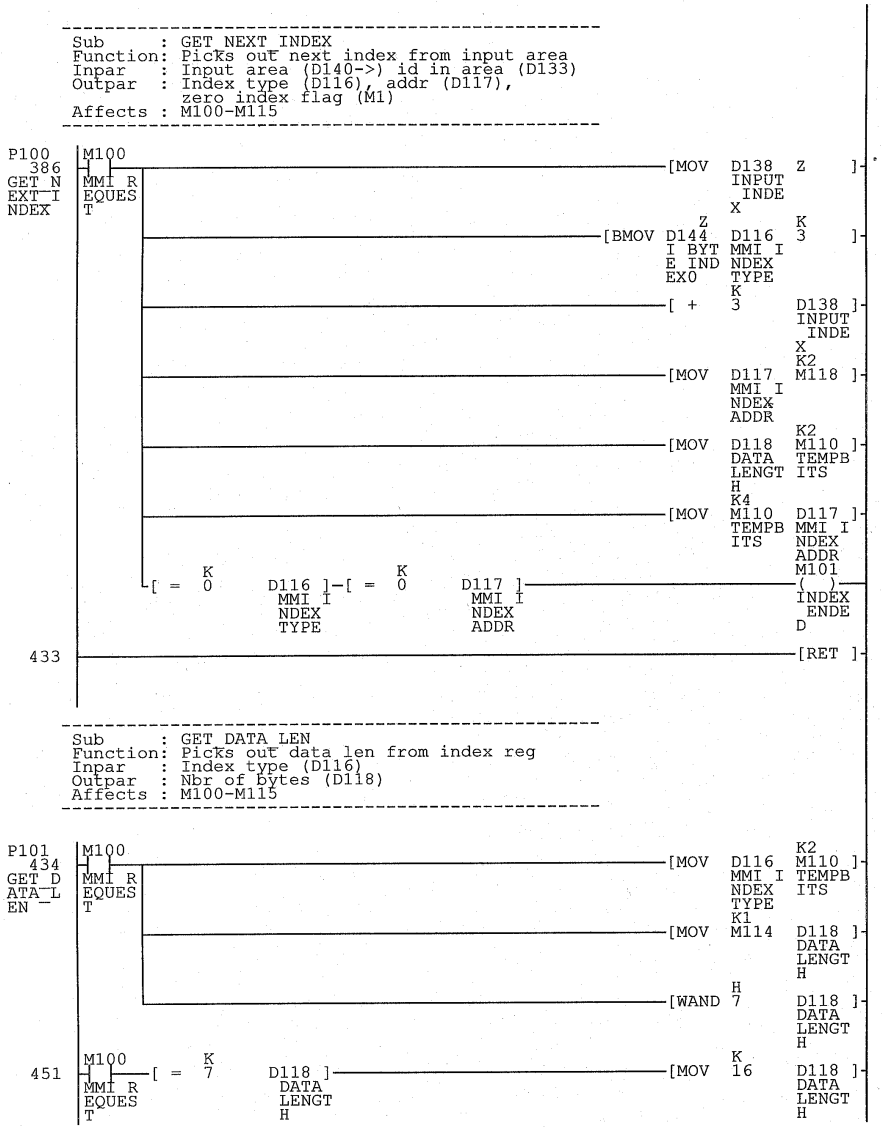

Convert registers to byte for easier handling
123
M100
MMI REQUEST
[SET M102]
[MOV K0 Z]
[MOV K0 V]
135
[FOR K16]
138
M100
MMI REQUEST
[MOV Z D120 I WORD CONTROL K4 M110 TEMPB ITS]
[MOV K2 M110 TEMPB ITS V D140 I BYTE CONTROL]
[MOV K2 M118 V D141 I BYTE DATA0]
[INC Z]
[+ K2 V]
162
[NEXT]
163
M100
MMI REQUEST
[MOV K0 V]
[MOV K0 D138 INPUT INDEX]
[FMOV K0 D180 O BYTE CONTROL K32]
Loop start : pick out index
P20
183
M100
MMI REQUEST
[CALL P100 GET NEXT INDEX]

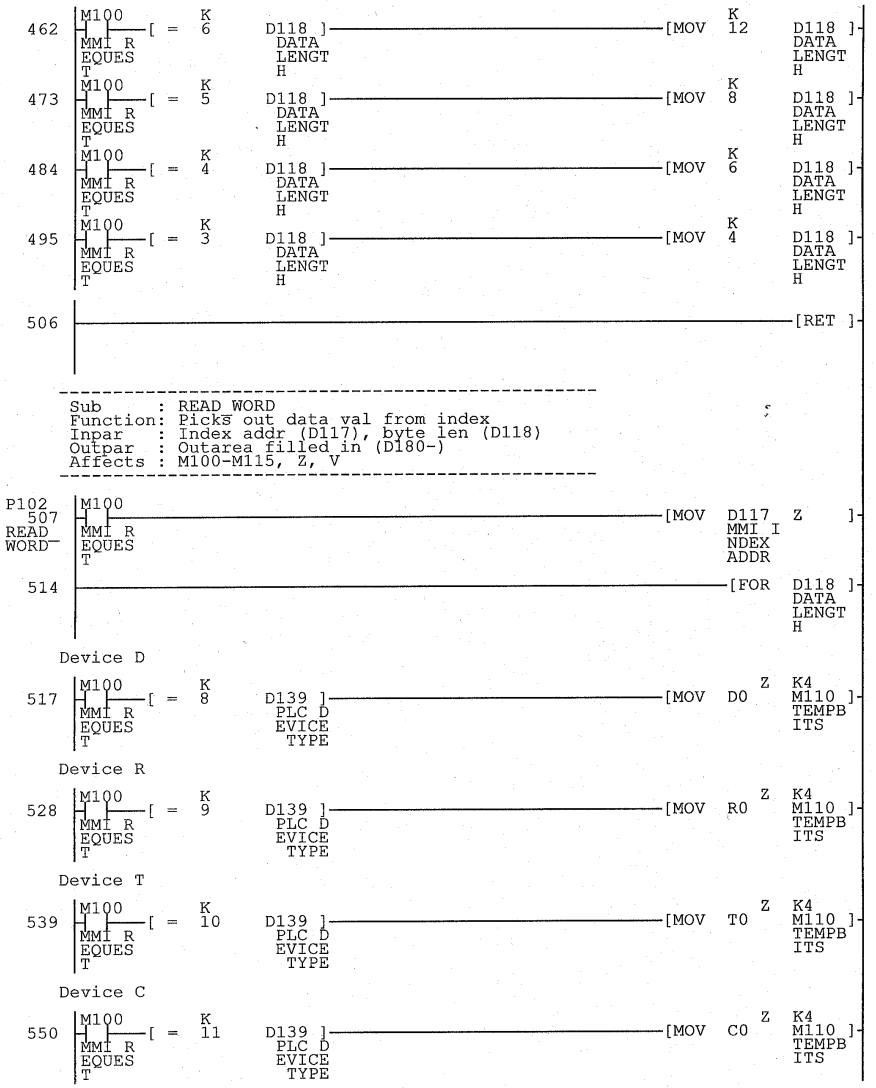

Jump out if index = 0
188
M100 MMI_R EQUEST
M101 INDEX ENDED
[CJ P2]
Pick out nbr of bytes to transfer
193
M100 MMI_R EQUEST
[CALL P101_GET_DATA_LEN]
Detect read/write and device type
197
M100 MMI_R EQUEST
[MOV D116 MMI_INDEX TYPE K1]
[MOV K2 M110 TEMPBITS]
[MOV M110 TEMPBITS D139 PLC_DEVICE_TYPE]
208
M113
M130 MMI_TYPE_WORD
210
M117
M131 MMI_WRITEINDEX
If words:Divide nbr of bytes=>nbr of words instead
212
M130 MMI_TYPE_WORD
[/ D118 DATA LENGTH K 2 D118 DATA LENGTH]
220
M131 MMI_WRITEINDEX
M130 MMI_TYPE_WORD
[CALL P102 READ_WORD]
225
M131 MMI_WRITEINDEX
M130 MMI_TYPE_WORD
[CALL P104 WRITE_WORD]
230
M131 MMI_WRITEINDEX
M130 MMI_TYPE_WORD
[CALL P105 WRITE_BIT]
235
M131 MMI_WRITEINDEX
M130 MMI_TYPE_WORD
[CALL P103 READ_BIT]
240
M100 MMI_REQUEST
[CJ P20]

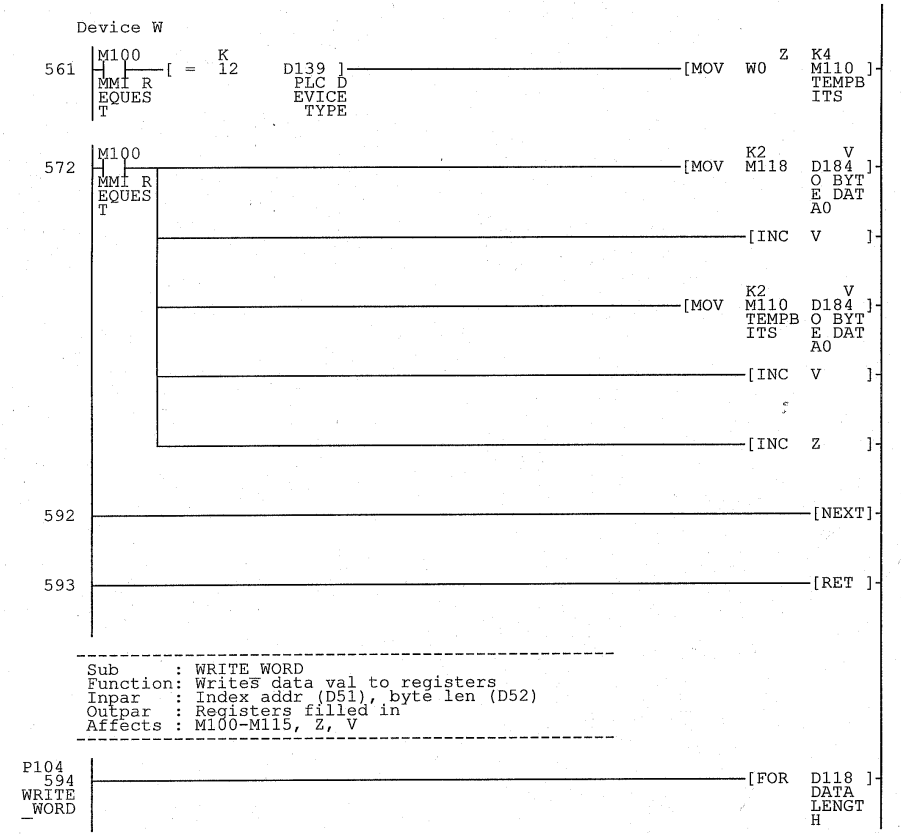

Convert bytes to registers again
P2
244
M100
MMI REQUEST
[MOV K 0 Z]
[MOV K 0 V]
256
[FOR K 16]
259
M100
MMI REQUEST
[MOV Z D180 K2 M110]
O BYTE CONTROL TEMPB ITS
[MOV Z D181 K2 M118]
O BYTE QDATA
[MOV K4 M110 V D100]
TEMPB ITS O WORD CTR LBYTE
[INC V]
[+ K 2 Z]
283
[NEXT]
Reply in response register
284
M100
MMI REQUEST
[MOV D178 Z]
CURRENT STN ID
[MOV Z D220 D100]
O WORD CTR LBYTE
Set COM-bit
295
M100
MMI REQUEST
[BSET D100 K 6]
O WORD CTR LBYTE
Take care of TOGGLE-bit
303
M100
MMI REQUEST
[WAND H 20 D120]
I WORD CONTROL
309
M100
MMI REQUEST
[= K 0 D120]
I WORD CONTROL
[BSET D100 K 5]
O WORD CTR LBYTE

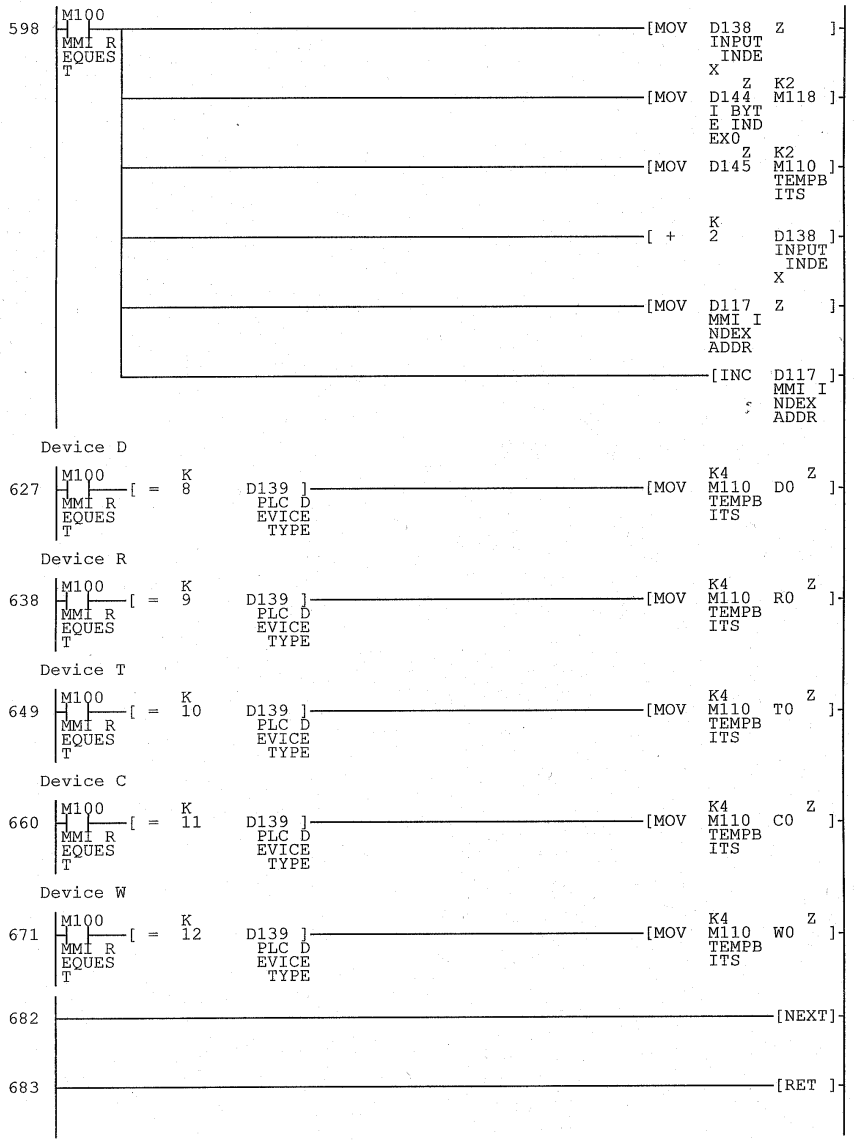

322
M100
MMI_R EQUEST
[<>
K 0
D120]
I_WOR D_CONTROL
[BRST
D100 O_WOR D_CTR LBYTE
K 5]
Now write data to the module
P0
335
M9036
[*
D178 CURRENT_STATION_ID
K 16
Z]
344
X0
XD
X1B
X1D
M102
[TO
H 0
K_Z 960
D100 O_WOR D_CTR LBYTE
K 16]
[MOV
D178 CURRENT_STATION_ID
Z]
[MOV
D100 O_WOR D_CTR LBYTE
Z D220]
[RST M102]
369
M102
[+
K 1
D178 CURRENT_STATION_ID]
[= D177 D178]
NBR_OF_STATIONS CURRENT_STATION_ID
[MOV
K 0
D178 CURRENT_STATION_ID]
385
[FEND]

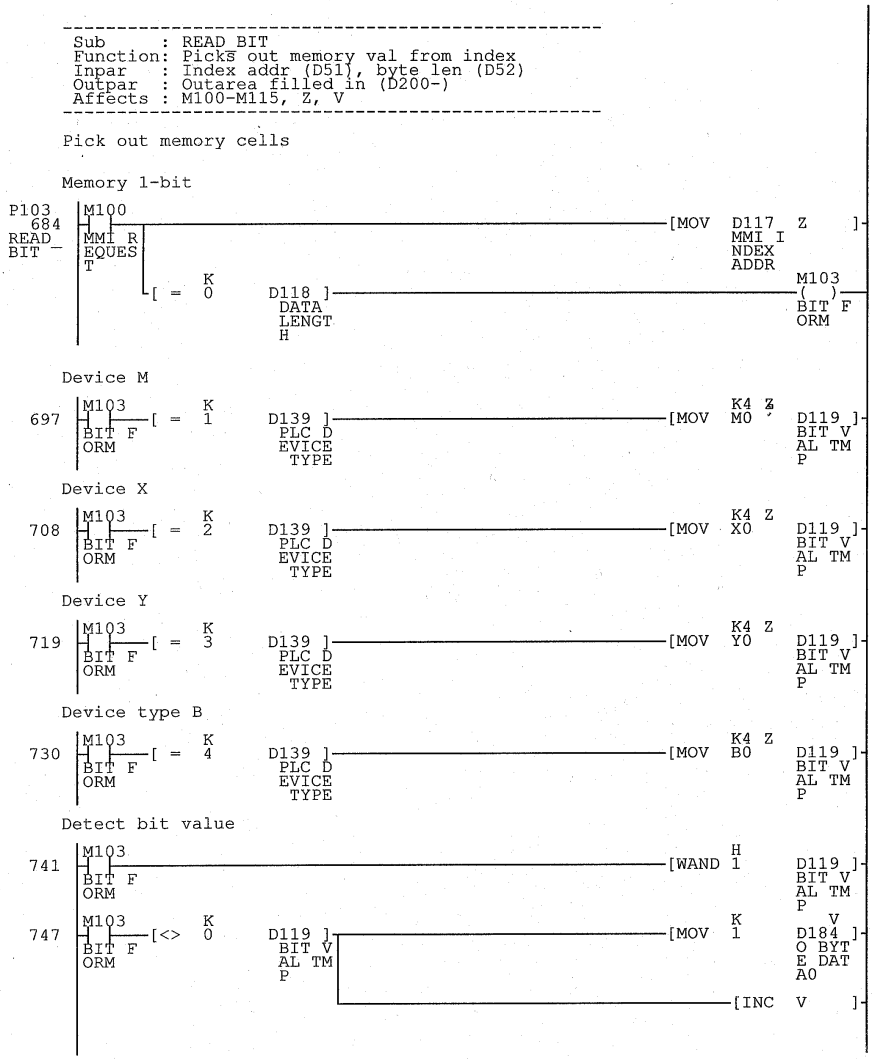
The subroutine section for Mitsubishis MELSEC A

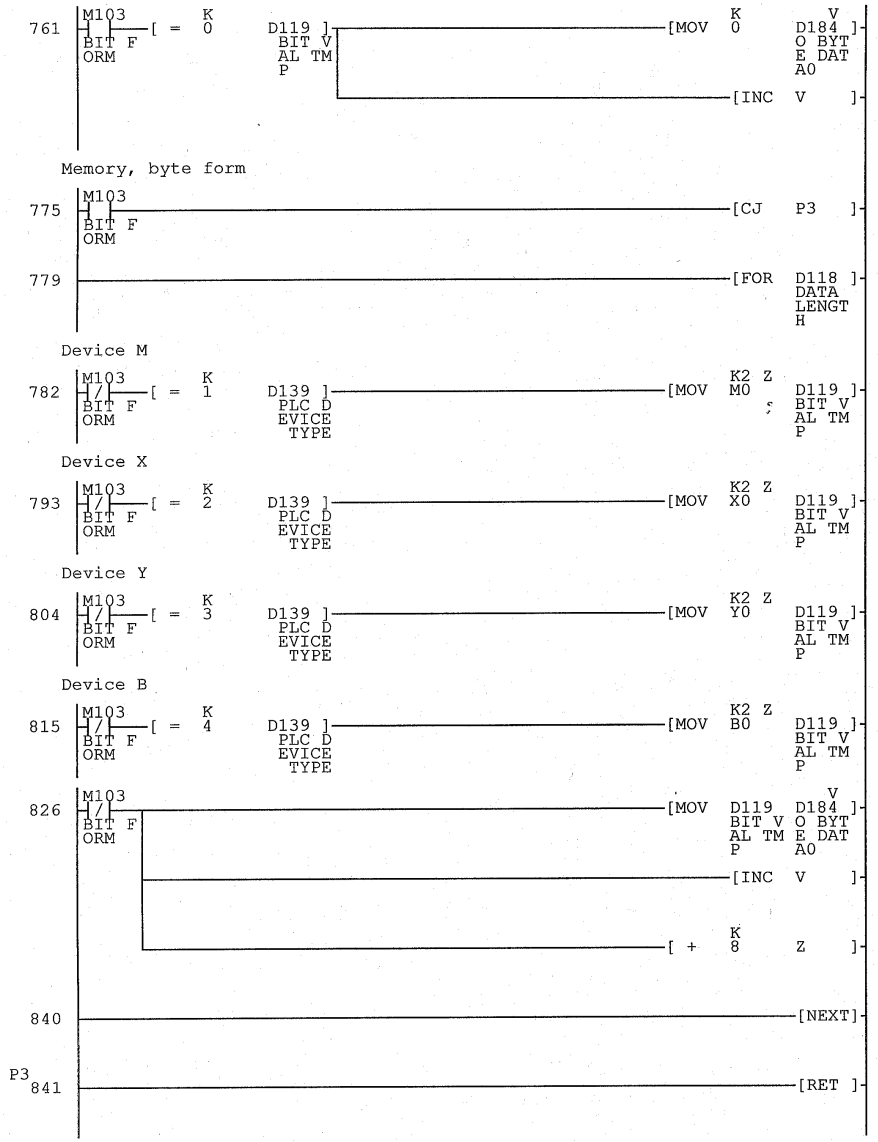


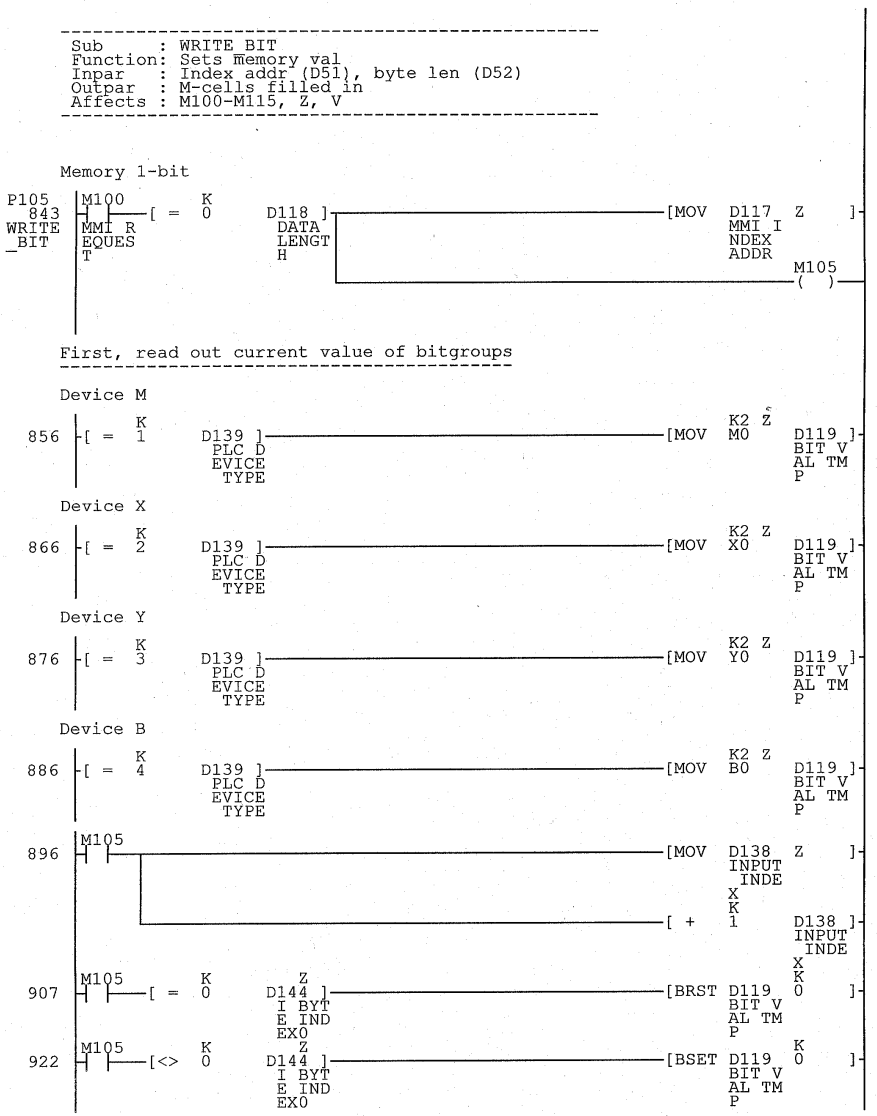

462 M100 MMI R EQUES T [= K 6 D118 DATA LENGTH] [MOV K 12 D118 DATA LENGTH]
473 M100 MMI R EQUES T [= K 5 D118 DATA LENGTH] [MOV K 8 D118 DATA LENGTH]
484 M100 MMI R EQUES T [= K 4 D118 DATA LENGTH] [MOV K 6 D118 DATA LENGTH]
495 M100 MMI R EQUES T [= K 3 D118 DATA LENGTH] [MOV K 4 D118 DATA LENGTH]
506 [RET]
Sub : READ WORD
Function: Picks out data val from index
Inpar : Index addr (D117), byte len (D118)
Outpar : Outarea filled in (D180-)
Affects : M100-M115, Z, V
P102 507 READ_WORD M100 MMI R EQUES T [MOV D117 MMI_INDEX ADDR Z]
514 [FOR D118 DATA LENGTH]
Device D
517 M100 MMI R EQUES T [= K 8 D139 PLC DEVICE TYPE] [MOV Z D0 K4 M110 TEMPBITS]
Device R
528 M100 MMI R EQUES T [= K 9 D139 PLC DEVICE TYPE] [MOV Z R0 K4 M110 TEMPBITS]
Device T
539 M100 MMI R EQUES T [= K 10 D139 PLC DEVICE TYPE] [MOV Z T0 K4 M110 TEMPBITS]
Device C
550 M100 MMI R EQUES T [= K 11 D139 PLC DEVICE TYPE] [MOV Z C0 K4 M110 TEMPBITS]

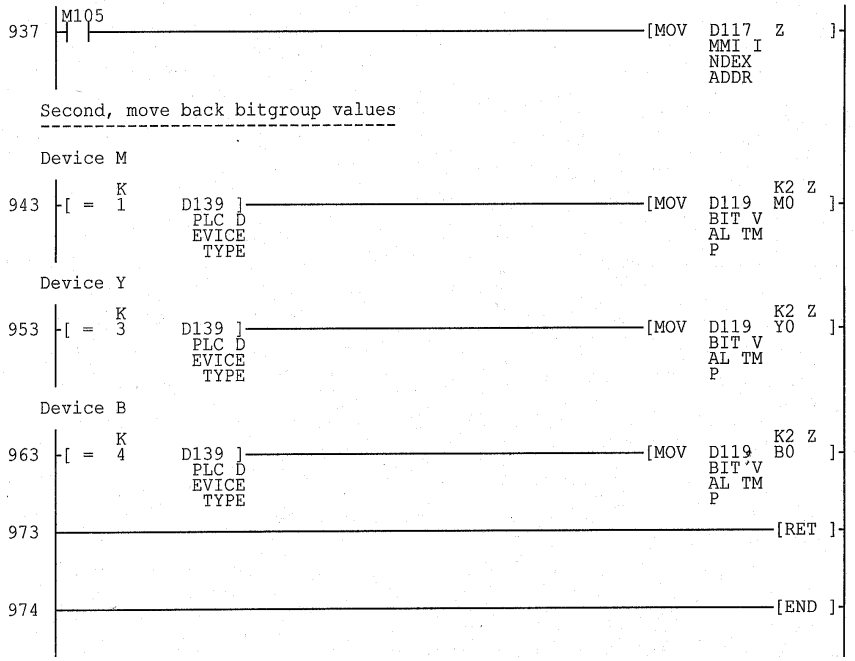

Device W
561
M100
MMI R EQUEST
K 12
D139] PLC D EVICE TYPE
[MOV W0 Z K4 M110] TEMPB ITS
572
M100
MMI R EQUEST
[MOV K2 M118 V D184] O BYT E DAT A0
[INC V]
[MOV K2 M110 TEMPB ITS V D184] O BYT E DAT A0
[INC V]
[INC Z]
592
[NEXT]
593
[RET]
Sub : WRITE_WORD
Function: Writes data val to registers
Inpar : Index addr (D51), byte len (D52)
Outpar : Registers filled in
Affects : M100-M115, Z, V
P104
594
WRITE _WORD
[FOR D118] DATA LENGT H


598 M100 MMI_REQUEST
[MOV D138 INPUT INDEX Z]
[MOV D144 I BYTE INDEX Z K2 M118]
[MOV D145 Z K2 M110 TEMPBITS]
[+ K 2 D138 INPUT INDEX]
[MOV D117 MMI INDEX ADDR Z]
[INC D117 MMI INDEX ADDR]
Device D
627 M100 MMI_REQUEST [= K 8 D139 PLC DEVICE TYPE] [MOV K4 M110 TEMPBITS Z D0]
Device R
638 M100 MMI_REQUEST [= K 9 D139 PLC DEVICE TYPE] [MOV K4 M110 TEMPBITS Z R0]
Device T
649 M100 MMI_REQUEST [= K 10 D139 PLC DEVICE TYPE] [MOV K4 M110 TEMPBITS Z T0]
Device C
660 M100 MMI_REQUEST [= K 11 D139 PLC DEVICE TYPE] [MOV K4 M110 TEMPBITS Z C0]
Device W
671 M100 MMI_REQUEST [= K 12 D139 PLC DEVICE TYPE] [MOV K4 M110 TEMPBITS Z W0]
682 [NEXT]
683 [RET]


Sub : READ_BIT
Function: Picks out memory val from index
Inpar : Index addr (D51), byte len (D52)
Outpar : Outarea filled in (D200-)
Affects : M100-M115, Z, V
Pick out memory cells
Memory 1-bit
P103 684 READ_BIT
M100
MMI_REQUEST
[= K 0 D118] DATA LENGTH
[MOV D117 Z] MMI_INDEX ADDR
M103 ( ) BIT FORM
Device M
697
M103 BIT FORM
[= K 1 D139] PLC DEVICE TYPE
[MOV K4 M0 Z] BIT VAL TMP
Device X
708
M103 BIT FORM
[= K 2 D139] PLC DEVICE TYPE
[MOV K4 X0 Z] BIT VAL TMP
Device Y
719
M103 BIT FORM
[= K 3 D139] PLC DEVICE TYPE
[MOV K4 Y0 Z] BIT VAL TMP
Device type B
730
M103 BIT FORM
[= K 4 D139] PLC DEVICE TYPE
[MOV K4 B0 Z] BIT VAL TMP
Detect bit value
741
M103 BIT FORM
[WAND H 1 D119] BIT VAL TMP
747
M103 BIT FORM
[<> K 0 D119] BIT VAL TMP
[MOV K 1 V D184] BYTE DATA A0
[INC V]


761
M103
BIT F
ORM
K
0
D119
BIT V
AL TM
P
MOV
K
0
V
D184
O BYT
E DAT
A0
INC
V
Memory, byte form
775
M103
BIT F
ORM
CJ
P3
779
FOR
D118
DATA
LENGT
H
Device M
782
M103
BIT F
ORM
K
1
D139
PLC D
EVICE
TYPE
MOV
K2 Z
M0
D119
BIT V
AL TM
P
Device X
793
M103
BIT F
ORM
K
2
D139
PLC D
EVICE
TYPE
MOV
K2 Z
X0
D119
BIT V
AL TM
P
Device Y
804
M103
BIT F
ORM
K
3
D139
PLC D
EVICE
TYPE
MOV
K2 Z
Y0
D119
BIT V
AL TM
P
Device B
815
M103
BIT F
ORM
K
4
D139
PLC D
EVICE
TYPE
MOV
K2 Z
B0
D119
BIT V
AL TM
P
826
M103
BIT F
ORM
MOV
D119
BIT V
AL TM
P
V
D184
O BYT
E DAT
A0
INC
V
+
K
8
Z
840
NEXT
P3
841
RET


Sub : WRITE BIT
Function: Sets memory val
Inpar : Index addr (D51), byte len (D52)
Outpar : M-cells filled in
Affects : M100-M115, Z, V
Memory 1-bit
P105 843 WRITE_BIT
M100 MMI_REQUEST
K 0
D118 DATA LENGTH
MOV D117 MMI_INDEX Z
MOV K1 ADDR
M105
First, read out current value of bitgroups
Device M
856
K 1
D139 PLC DEVICE TYPE
MOV K2 M0 D119 BIT VAL TMP
Device X
866
K 2
D139 PLC DEVICE TYPE
MOV K2 X0 D119 BIT VAL TMP
Device Y
876
K 3
D139 PLC DEVICE TYPE
MOV K2 Y0 D119 BIT VAL TMP
Device B
886
K 4
D139 PLC DEVICE TYPE
MOV K2 B0 D119 BIT VAL TMP
896
M105
MOV D138 INPUT INDEX Z
+ D138 INPUT INDEX K1
907
M105
K 0
D144 I BYTE IND EX0 Z
BRST D119 BIT VAL TMP K0
922
M105
K 0
D144 I BYTE IND EX0 Z
BSET D119 BIT VAL TMP K0


937 M105
[MOV D117 Z MMI INDEX ADDR]
Second, move back bitgroup values
Device M
943 [= K 1 D139 PLC DEVICE TYPE]
[MOV D119 BIT VAL TM P K2 Z M0]
Device Y
953 [= K 3 D139 PLC DEVICE TYPE]
[MOV D119 BIT VAL TM P K2 Z Y0]
Device B
963 [= K 4 D139 PLC DEVICE TYPE]
[MOV D119 BIT VAL TM P K2 Z B0]
973 [RET]
974 [END]

The PLC program section for SIMATIC S5

```
FB 190                                D:TESTARST.S5D                    LEN=168
                                                                         Page
Segment  1                      MAIN PROFILE HANDLER


------------------------------------------------------------------------------
This function block is the main handler of the MMI profile container

The function block handles one complete container in one PLC scan. If read ind
found, FB191 is called. If write index found, FB192 is called
------------------------------------------------------------------------------
Name :PROFILE
Decl :LEN        I/Q/D/B/T/C: D  KM/KH/KY/KS/KF/KT/KC/KG: KF
Decl :READ       I/Q/D/B/T/C: D  KM/KH/KY/KS/KF/KT/KC/KG: KF
Decl :WRI        I/Q/D/B/T/C: D  KM/KH/KY/KS/KF/KT/KC/KG: KF
Decl :INT        I/Q/D/B/T/C: I  BI/BY/W/D: BY
Decl :HERR       I/Q/D/B/T/C: Q  BI/BY/W/D: BY
Decl :TEMS       I/Q/D/B/T/C: T

0017       :LW   =READ
0018       :T    FW 220                        Incontainer start address
0019       :T    FW 230
001A       :DO   FW 220
001B       :L    IB   0                        Read control byte from incont.
001C       :T    FY 206                        Store it
001D       :LW   =WRI
001E       :T    FW 222                        Outcontainer start address
001F       :T    FW 232
0020       :L    =INT
0021       :T    FY 240
0022       :L    KF +0
0024       :T    =HERR                         Clear error code
0025       :A    F  240.0
0026       :R    F  240.0                      Clear error flag
0027       :A    F  206.6                      Terminal present ?
0028       :JC   =TPRE
0029       :L    KF +1                         Error 1 : Comm error
002B       :JU   =ERR
002C       :
002D TPRE  :A    F  206.5                      Does the terminal toggle TGL?
002E       :L    KT 002.2
0030       :SD   =TEMS                         Timeout timer
0031       :AN   =TEMS
0032       :JC   =YTGL
0033       :L    KF +2                         Error 2 : Terminal not in RUN
0035       :JU   =ERR
0036       :
0037 YTGL  :A    F  206.7                      Test if new container available
0038       :A    F  240.7                      Compare old with new
0039       :O
003A       :AN   F  206.7
003B       :AN   F  240.7
003C       :JC   =END                          No new container
003D       :
003E       :A    F  206.7                      Pulse
003F       :=    F  240.7
0040       :
0041       :L    FW 222                        First outcontainer pos : 4
0042       :L    KF +4
0044       :+F
0045       :T    FW 222
0046       :
```

```
FB 190                              D:TESTARST.S5D                    LEN=168
                                                                       Page
0047      :L   FW 220                          First incontainer pos : 4
0048      :L   KF +4
004A      :+F
004B      :T   FW 220
004C LOOP :
004D      :DO  FW 220
004E      :L   IB   0                          Read first index byte
004F      :T   FY 206                          Store it
0050      :L   FY 206
0051      :L   KF +0                           Was it 0 ?
0053      :!=F
0054      :JC  =END                            Then ,no more indexes
0055      :AN  F  206.7                        Was it a read index ?
0056      :JC  FB 191                          Then call read index function
0057 Name :READ IND
0058 READ :    FW 220
0059 WRI  :    FW 222
005A NBR  :    FY 224
005B      :L   FY 224                          Check return value from FB191
005C      :L   KF +0
005E      :!=F
005F      :JC  =OKRT                           Was there an error return ?
0060      :AN  F  240.0
0061      :S   F  240.0                        Then, set error bit
0062      :L   FY 224
0063      :JU  =ERR
0064      :
0065 OKRT :AN  F  206.7                        Was it a write index ?
0066      :JC  =CONT
0067      :JU  FB 192                          Then call write index function
0068 Name :WRITE ID
0069 READ :    FW 220
006A NBR  :    FY 224
006B      :L   FY 224                          Check return value from FB192
006C      :L   KF +0
006E      :!=F
006F      :JC  =CONT                           Was there an error return ?
0070      :AN  F  240.0
0071      :S   F  240.0                        Then, set error bit
0072      :L   FY 224
0073      :JU  =ERR
0074      :
0075 CONT :L   FW 220                          Check if incontainer done..
0076      :L   FW 230
0077      :-F
0078      :LW  =LEN
0079      :<F
007A      :A(
007B      :L   FW 222              01          .. or that outcontainer is done
007C      :L   FW 232              01
007D      :-F                      01
007E      :LW  =LEN                01
007F      :<F                      01
0080      :)                       01
0081      :JC  =LOOP
0082      :JU  =END
0083      :
0084 ERR  :T   =HERR                           Error handling: Store err code
0085 FILL :L   KF +0                           Fill container with zeros
0087      :DO  FW 222
```

```
FB 190                                  D:TESTARST.S5D                  LEN=168
                                                                         Page

0088        :T    QB    0
0089        :L    KF +1
008B        :L    FW 222
008C        :+F
008D        :T    FW 222
008E        :L    FW 232
008F        :-F
0090        :LW   =LEN
0091        :<F
0092        :JC   =FILL
0093        :
0094 END    :DO   FW 230
0095        :L    IB    0                   Get ctrl byte from incontainer
0096        :T    FY 206
0097        :AN   F  206.6
0098        :S    F  206.6                  Set the COM-bit
0099        :A    F  240.0
009A        :=    F  206.4                  Toggle the TGL-bit
009B        :L    FY 206
009C        :DO   FW 232
009D        :T    QB    0                   Write ctrl byte to outcontainer
009E        :T    FY 206
009F        :
00A0        :L    FY 240
00A1        :T    =INT
00A2        :BE
```

```
FB 191                                  D:TESTARST.S5D                  LEN=245
                                                                         Page
Segment  1                        READ INDEX

------------------------------------------
Calculate pointers and byte size
------------------------------------------
Name :READ IND
Decl :READ       I/Q/D/B/T/C: I  BI/BY/W/D: W
Decl :WRI        I/Q/D/B/T/C: Q  BI/BY/W/D: W
Decl :NBR        I/Q/D/B/T/C: Q  BI/BY/W/D: BY

000E      :L   KF +0
0010      :T   =NBR                              Clear return value
0011      :L   =READ
0012      :T   FW 200                            Incontainer start address
0013      :L   KF +3
0015      :+F
0016      :T   =READ                             Point to next index
0017      :L   =WRI
0018      :T   FW 202                            Outcontainer start address
0019      :DO  FW 200
001A      :L   IB   0                            Get HIGH byte of index
001B      :T   FW 206                            Store it
001C      :L   FW 200
001D      :L   KF +1                             Point to next byte
001F      :+F
0020      :T   FW 200
0021      :DO  FW 200
0022      :L   IW   0                            Get MIDDLE+LOW (=device address
0023      :T   FW 208                            Store it in 208-209
0024      :L   FW 200
0025      :L   KF +2
0027      :+F
0028      :T   FW 200                            Point to next index
0029      :L   FY 207                            Calculate how many bytes
002A      :SRW     4
002B      :L   KH 0007
002D      :AW
002E      :T   FY 212
002F      :A(
0030      :AN  F  212.2             01
0031      :AN  F  212.1             01
0032      :)                        01
0033      :O
0034      :A(
0035      :AN  F  212.2             01
0036      :A   F  212.1             01
0037      :AN  F  212.0             01
0038      :)                        01
0039      :JC  =END
003A      :A   F  212.2
003B      :JC  =TST6
003C      :L   KF +4                             Length is 4 bytes
003E      :JU  =STOR
003F TST6 :A   F  212.1
0040      :O   F  212.0
0041      :JC  =TST8
0042      :L   KF +6                             Length is 6 bytes
0044      :JU  =STOR
0045 TST8 :A   F  212.1
0046      :JC  =TS12
```

```
FB 191                                  D:TESTARST.S5D                  LEN=245
                                                                         Page
0047      :L   KF +8                                Length is 8 bytes
0049      :JU  =STOR
004A TS12 :A   F  212.0
004B      :JC  =TS16
004C      :L   KF +12                               Length is 12 bytes
004E      :JU  =STOR
004F      :
0050 TS16 :L   KF +16                               Length is 16 bytes
0052 STOR :T   FY 212                               Store length
0053 END  :
0054      :***


Segment  2

-------------------------------------------
Handle reading of bit devices (F/Q/I)
-------------------------------------------
0055      :L   KF +0
0057      :T   FY 250                               Reset device type flags
0058      :O   F  207.0                             Test if input device
0059      :ON  F  207.1
005A      :O   F  207.2
005B      :O   F  207.3
005C      :JC  =TSTQ
005D      :S   F  250.0                             Read inputs is selected
005E      :JU  =STRT
005F TSTQ :ON  F  207.0                             Test if output device
0060      :ON  F  207.1
0061      :O   F  207.2
0062      :O   F  207.3
0063      :JC  =TSTM
0064      :S   F  250.1                             Read outputs is selected
0065      :JU  =STRT
0066 TSTM :ON  F  207.0                             Test if memory device
0067      :O   F  207.1
0068      :O   F  207.2
0069      :O   F  207.3
006A      :JC  =M001                                No bits selected, jump out
006B      :S   F  250.2                             Read memory is selected
006C STRT :L   FW 202                               Outcontainer pointer
006D      :T   FW 216
006E      :O   F  207.6
006F      :O   F  207.5
0070      :O   F  207.4
0071      :JC  =BYTE                                Jump if more than one bit
0072      :
0073      :                                         The following handles 1-bit rea
0074      :
0075      :L   FY 211
0076      :SLW      8
0077      :L   FY 209
0078      :OW
0079      :T   FW 214
007A      :AN  F  250.0                             Input bit ?
007B      :JC  =M030
007C      :DO  FW 214
007D      :A   I    0.0                             Get input bit
007E      :JC  =M003
007F      :JU  =M040
```

```
FB 191                                  D:TESTARST.S5D                  LEN=245
                                                                         Page
0080      :
0081 M030 :AN   F  250.1                          Output bit ?
0082      :JC   =M031
0083      :DO   FW 214
0084      :A    Q    0.0                          Get output bit
0085      :JC   =M003
0086      :JU   =M040
0087 M031 :DO   FW 214
0088      :A    F    0.0                          Get flag bit
0089      :JC   =M003
008A      :JU   =M040
008B M040 :L    KF +0                             Bit value OFF
008D      :JU   =M004
008E M003 :L    KF +1                             Bit value ON
0090 M004 :
0091      :DO   FW 202
0092      :T    QB   0                            Write bit value to outcontainer
0093      :JU   =M005
0094      :
0095      :                                       The following handles byte read
0096      :
0097 BYTE :L    FW 208                            Device address from index
0098      :SRW     3                              Divide by 8
0099      :T    FW 208
009A LOOP :AN   F  250.0                          Is it input byte ?
009B      :JC   =M050
009C      :DO   FW 208
009D      :L    IB   0                            Get input byte
009E      :JU   =M060
009F M050 :AN   F  250.1                          Is it output byte ?
00A0      :JC   =M051
00A1      :DO   FW 208
00A2      :L    QB   0                            Get output byte
00A3      :JU   =M060
00A4 M051 :DO   FW 208
00A5      :L    FY   0                            Get memory byte
00A6 M060 :DO   FW 202
00A7      :T    QB   0                            Store in outcontainer
00A8      :L    FW 208
00A9      :L    KF +1                             Increment data source pointer
00AB      :+F
00AC      :T    FW 208
00AD      :L    FW 202
00AE      :L    KF +1                             Increment data dest pointer
00B0      :+F
00B1      :T    FW 202
00B2      :L    FW 216
00B3      :-F
00B4      :L    FY 212                            All bytes read ?
00B5      :<F
00B6      :JC   =LOOP                             Go for next byte...
00B7 M005 :L    FW 202
00B8      :T    =WRI
00B9 M001 :
00BA      :***

Segment  3                    READ DATABASE

------------------------------------------------
```

```
FB 191                                  D:TESTARST.S5D                  LEN=245
                                                                         Page
-------------------------------------------
Handle reading of data blocks (DB)
-------------------------------------------
00BB       :O   F  207.0                    Test if D
00BC       :O   F  207.1
00BD       :O   F  207.2
00BE       :ON  F  207.3
00BF       :JC  =M001
00C0       :
00C1       :L   FW 202                      Outcontainer pointer
00C2       :T   FW 216
00C3       :
00C4       :
00C5       :L   FY 208                      Datablock number to use
00C6       :T   FW 210
00C7       :DO  FW 210
00C8       :C   DB   0                      Open datablock
00C9       :L   FY 209                      Dataword in block
00CA       :T   FW 210
00CB       :A   F  213.0                    Flag to select high/low byte
00CC       :R   F  213.0
00CD M005  :A   F  213.0
00CE       :JC  =M002
00CF       :DO  FW 210
00D0       :L   DL   0                      Get low byte fron datablock
00D1       :JU  =M003
00D2 M002  :DO  FW 210
00D3       :L   DR   0                      Get high byte from datablock
00D4 M003  :AN  F  213.0
00D5       :=   F  213.0
00D6       :
00D7       :DO  FW 202
00D8       :T   QB   0                      Put data in outcontainer
00D9       :
00DA       :A   F  213.0
00DB       :JC  =M004
00DC       :L   FW 210                      Increase data source ptr
00DD       :L   KF +1
00DF       :+F
00E0       :T   FW 210
00E1       :
00E2 M004  :L   FW 202
00E3       :L   KF +1                       Increase data dest ptr
00E5       :+F
00E6       :T   FW 202
00E7       :L   FW 216
00E8       :-F
00E9       :L   FY 212
00EA       :<F
00EB       :JC  =M005                       Do all bytes
00EC       :L   FW 202
00ED       :T   =WRI
00EE M001  :
00EF       :BE
```

```
FB 192                                  D:TESTARST.S5D                  LEN=216
                                                                         Page
Segment  1                          WRITE INDEX

------------------------------------------
Calculate pointers and data size
------------------------------------------
Name :WRITE ID
Decl :READ       I/Q/D/B/T/C: I  BI/BY/W/D: W
Decl :NBR        I/Q/D/B/T/C: Q  BI/BY/W/D: BY

000B      :L   KF +0                            Clear return value
000D      :T   =NBR
000E      :L   =READ
000F      :T   FW 200                           Incontainer start address
0010      :L   KF +3
0012      :+F
0013      :T   =READ                            Ptr to next index
0014      :DO  FW 200
0015      :L   IB   0                           Get HIGH byte of index
0016      :T   FW 206
0017      :L   FW 200
0018      :L   KF +1                            Point to next index byte
001A      :+F
001B      :T   FW 200
001C      :DO  FW 200
001D      :L   IW   0                           Device address
001E      :T   FW 208
001F      :
0020      :L   FW 200                           Point to the data to write
0021      :L   KF +2
0023      :+F
0024      :T   FW 200
0025      :
0026      :L   FY 207
0027      :SRW     4
0028      :L   KH 0007
002A      :AW
002B      :T   FY 212                           This is the size from index
002C      :A(
002D      :AN  F  212.2             01
002E      :AN  F  212.1             01
002F      :)                        01
0030      :O
0031      :A(
0032      :AN  F  212.2             01
0033      :A   F  212.1             01
0034      :AN  F  212.0             01
0035      :)                        01
0036      :JC  =END
0037      :A   F  212.2
0038      :JC  =TST6
0039      :L   KF +4                            4 bytes
003B      :JU  =STOR
003C TST6 :A   F  212.1
003D      :O   F  212.0
003E      :JC  =TST8
003F      :L   KF +6                            6 bytes
0041      :JU  =STOR
0042 TST8 :A   F  212.1
0043      :JC  =TS12
0044      :L   KF +8                            8 bytes
```

```
FB 192                                  D:TESTARST.S5D                  LEN=216
                                                                         Page
0046      :JU  =STOR
0047 TS12 :A   F  212.0
0048      :JC  =TS16
0049      :L   KF +12                                 12 bytes
004B      :JU  =STOR
004C TS16 :L   KF +16                                 16 bytes
004E STOR :T   FY 212                                 Store size
004F END  :
0050      :***


Segment  2

-------------------------------------------
Handle writing of bit devices (Q/I/F)
-------------------------------------------
0051      :L   KF +0
0053      :T   FY 250                                 Reset devicetype flags
0054      :ON  F  207.0                               Test for output
0055      :ON  F  207.1
0056      :O   F  207.2
0057      :O   F  207.3
0058      :JC  =TSTM
0059      :S   F  250.0                               Output selected
005A      :JU  =STRT
005B TSTM :ON  F  207.0                               Test for memory
005C      :O   F  207.1
005D      :O   F  207.2
005E      :O   F  207.3
005F      :JC  =END
0060      :S   F  250.1                               Memory selected
0061 STRT :L   FW 200
0062      :T   FW 216
0063      :O   F  207.6
0064      :O   F  207.5
0065      :O   F  207.4
0066      :JC  =BYTE                                  Jump if more than one bit
0067      :DO  FW 200
0068      :L   IB   0                                 Get new data (ON/OFF) form inco
0069      :T   FY 213
006A      :L   FW 208                                 Get device address
006B      :SRW      3                                 Divide by 8
006C      :T   FY 211                                 This is the byte nbr
006D      :L   FW 208
006E      :L   KH 0007
0070      :AW
0071      :T   FY 210                                 This is the bit nbr
0072      :AN  F  250.0
0073      :JC  =ISM
0074      :A   F  213.0
0075      :DO  FW 210
0076      :=   Q    0.0                               Write data to output
0077      :JU  =DONE
0078 ISM  :A   F  213.0
0079      :DO  FW 210
007A      :=   F    0.0                               Write data to memory
007B      :JU  =DONE
007C      :
007D      :                                           The following handles byte writ
007E      :
```

```
FB 192                                  D:TESTARST.S5D                  LEN=216
                                                                         Page
007F BYTE :L   FW 208                                 Get device address
0080      :SRW      3                                 Divide by 8
0081      :T   FW 208
0082 LOOP :DO  FW 200
0083      :L   IB   0                                 Read input data
0084      :AN  F  250.0
0085      :JC  =SETM
0086      :DO  FW 208
0087      :T   QB   0                                 Store data in outputs
0088      :JU  =NEXT
0089 SETM :DO  FW 208
008A      :T   FY   0                                 Store data in memory
008B NEXT :L   FW 208
008C      :L   KF +1
008E      :+F
008F      :T   FW 208                                 Increase data dest ptr
0090      :L   FW 200
0091      :L   KF +1
0093      :+F
0094      :T   FW 200                                 Increase data src ptr
0095      :L   FW 216
0096      :-F
0097      :L   FY 212                                 All bytes done ?
0098      :<F
0099      :JC  =LOOP                                  Continue until completed
009A DONE :L   FW 200
009B      :T   =READ
009C END  :
009D      :***


Segment  3                    WRITE DB

------------------------------------------
Handle writing of DB-data
------------------------------------------
009E      :O   F  207.0                               Test if D
009F      :O   F  207.1
00A0      :O   F  207.2
00A1      :ON  F  207.3
00A2      :JC  =END
00A3      :L   FY 208
00A4      :T   FW 210                                 Datablock number to use
00A5      :DO  FW 210
00A6      :C   DB   0                                 Open the data base
00A7      :L   FY 209                                 Dataword in datablock
00A8      :T   FW 210
00A9      :L   FW 200
00AA      :T   FW 216                                 Input data ptr
00AB      :A   F  213.0                               Flag to select high/low byte
00AC      :R   F  213.0
00AD      :
00AE LOOP :DO  FW 200
00AF      :L   IB   0                                 Get input data
00B0      :A   F  213.0
00B1      :JC  =M002
00B2      :DO  FW 210
00B3      :T   DL   0                                 Store low byte
00B4      :
00B5      :JU  =M003
```

```
FB 192                                  D:TESTARST.S5D                  LEN=216
                                                                         Page
00B6 M002 :DO  FW 210
00B7      :T   DR   0                                Store high byte
00B8      :
00B9 M003 :AN  F  213.0
00BA      :=   F  213.0
00BB      :
00BC      :A   F  213.0
00BD      :JC  =M004
00BE      :L   FW 210                                Increase data dest ptr
00BF      :L   KF +1
00C1      :+F
00C2      :T   FW 210
00C3      :
00C4 M004 :L   FW 200                                Increase data src ptr
00C5      :L   KF +1
00C7      :+F
00C8      :T   FW 200
00C9      :L   FW 216
00CA      :-F
00CB      :L   FY 212                                Check that all bytes done
00CC      :<F
00CD      :JC  =LOOP
00CE      :L   FW 200
00CF      :T   =READ
00D0      :L   FY 212
00D1 END  :
00D2      :BE
```

The PLC program section for SIMATIC S7

SIMATIC ...ion 1\CPU315-2DP1\S7 Program(3)\AP-off\FB110 01/22/1997 14:22:25

FB110 - <Off-line>

Name: **Family:**
Author: **Version:** 00.00
Time stamp Code: DT#1996-12-10-16:14:57.000
Interface: DT#1996-12-09-09:53:29.000
Length (Block / MC7 Code / Data): 00560 00424 00008

Address	Decl.	Symbol	Data Type	Initial Value	Comment
0.0	in	fb_in_offset	DWORD	DW#16#0	
4.0	in	fb_out_offset	DWORD	DW#16#0	
8.0	in	con_len	WORD	W#16#0	
10.0	in	db_address	WORD	W#16#0	
	out				
	in_out				
12.0	stat	out_last_cycle	BYTE	B#16#0	
13.0	stat	out_this_cycle	BYTE	B#16#0	
14.0	stat	m_110	BYTE	B#16#0	
	temp				

This is the main handler of the MMI profile container

Parameters : fb_in_offset First byte in the input container
fb_out_offset First byte in the output container
db_address Number of the database to use
con_len Length of container in bytes (min.32)

Example : The terminal is configured to be on input and output byte 64,
32 bytes allocated and database to communcate with is 51:
fb_in_offset = 64
fb_out_offset = 64
db_address = 51
con_len = 32

The function block handles one complete container in one scan. If read index, FC111 is called.
If write index, FC112 is called.
Note that the control byte is not set until next scan, to avoid timing problems

```
Network: 1       COM_E700
      L     #fb_in_offset              // Incontainer start address
      T     MW   220                   // Remember incontainer start
      T     MW   230
//
      L     #fb_in_offset              // Incontainer start address
      SLD   3                          //
      LAR1
      L     EB [AR1,P#0.0]             // Get the proper control byte from inp.container
      T     MB   206                   // Store it
//
      L     #fb_out_offset             // Outcontainer start address
      T     MW   222                   // Store it
      T     MW   232
//
      L     #m_110                     // Previous input control byte
      T     MB   240                   // Store it
//
      L     0
      T     MB   111                   // Clear error code
//
      U     M    240.0
      R     M    240.0                 // Clear error flag
//
      U     M    206.6                 // Terminal present ?
      SPB   WEI1                       //
      L     1                          // Error 1 : Comm error
      SPA   FEH                        //
//
WEI1: U     M    206.5                 // Does the terminal toggle the toggle bit ?
      UN    M    206.5
      L     S5T#2S                     // Timeout timer
      SE    T     99
      UN    T     99
      SPB   WEI2
      L     2                          // Error 2 : Terminal not in RUN
      SPA   FEH
//
WEI2: U     M    206.7                 // Test if new container
      U     M    240.7                 // Compare old container byte with the new one
      O
      UN    M    206.7
```

```
SIMATIC              ...ion 1\CPU315-2DP1\S7 Program(3)\AP-off\FB110 01/22/1997 14:22:25
      UN    M    240.7
      SPB   ENDE                        // No new container
//
      U     M    206.7                  // Pulse
      =     M    240.7
//
      L     MW   222                    // Outcontainer pointer
      L     4
      +I
      T     MW   222                    // First data position : 4
//
      L     MW   220                    // Incontainer pointer
      L     4
      +I
      T     MW   220                    // First index position : 4
//
LOOP: NOP   0
      L     MW   220                    // Incontainer pointer
      ITD
      SLD   3
      T     MD   242                    // Make it double word
      L     EB [MD 242]                 // Get first index byte
      T     MB   206                    // Store it
      L     MB   206
      L     0
      ==I
      SPB   DONE                        // Jump out if index is 0
//
      U     M    206.7                  // Is it a read index ?
      SPB   ANK1
      CALL  FC   111                    // Then call read index function
       DataBase:=#db_address
ANK1: NOP   0
      L     MB   224                    // Check return value from FC111
      L     0
      ==I
      SPB   WEI3                        // Was there an error return ?
      UN    M    240.0
      S     M    240.0                  // Then, set error bit
      L     MB   224
      SPA   FEH                         // Jump to error
WEI3: UN    M    206.7                  // Is it a write index ?
      SPB   WEI4
      CALL  FC   112                    // Then call write index function
       DataBase:=#db_address
      L     MB   224                    // Check return value from FC112
      L     0
      ==I
      SPB   WEI4                        // Was there an error return ?
      UN    M    240.0
      S     M    240.0                  // Then set error bit
      L     MB   224                    //
      SPA   FEH                         // Jump to error
WEI4: L     MW   220                    // Check if incontainer is done
      L     MW   230
      -I
      L     #con_len
      <I
      U(
      L     MW   222                    // .. or that outcontainer is done
      L     MW   232
      -I
      L     #con_len
      <I
      )
      SPB   LOOP                        // Continue id conatiner not completed
DONE: L     #fb_in_offset               // Container done !
      SLD   3
      LAR1
      L     EB [AR1,P#0.0]              // Get input control byte
//
      T     MB   206                    // Store it
      SPA   ENDE
//-------  ERROR HANDLING  ---------
FEH:  T     MB   111                    // Store error code
FEH1: L     MW   222
      ITD
      SLD   3
      T     MD   242
//
```

```
SIMATIC              ...ion 1\CPU315-2DP1\S7 Program(3)\AP-off\FB110 01/22/1997 14:22:25
      L     0                              // Write controlbyte to output container
      T     AB [MD 242]
//
      L     1
      L     MW   222
//
      +I
      T     MW   222
//
      L     MW   232
      -I                                   // INTERNES MERKERBYTE SICHERN
//
      L     32
      <I
//
      SPB   FEH1
ENDE: NOP   0
      NOP   0
      UN    M    206.6
      S     M    206.6                     // Set the COM-bit
      UN    M    206.5
      =     M    206.5                     // Toggle the Toggel-bit
      U     M    240.0
      =     M    206.4
//----------------------
      L     MB   206                       // Delay answer one scan to avoid timing problems
      T     #out_this_cycle
      L     #fb_out_offset
      SLD   3
      LAR1
      L     #out_last_cycle
      T     AB [AR1,P#0.0]
      L     #out_this_cycle
      T     #out_last_cycle
      L     MB   240
      T     #m_110
      NOP   0
```

SIMATIC ...ion 1\CPU315-2DP1\S7 Program(3)\AP-off\FC111 02/12/1997 12:57:52

FC111 - <Off-line>

Name: LESE_IND **Family:**
Author: Sia **Version:** 00.01
Time stamp Code: DT#1997-02-06-10:12:53.000
Interface: DT#1996-12-05-10:00:32.000
Length (Block / MC7 Code / Data): 00888 00724 00002

Address	Decl.	Symbol	Data Type	Initial Value	Comment
0.0	in	DataBase	WORD	W#16#0	
	out				
	in_out				
0.0	temp	IsX	BOOL		
0.1	temp	IsY	BOOL		
0.2	temp	IsM	BOOL		

This function handles one read index

Network: 1 Index handling

This network picks out index and converts size to read

```
      L     0
      T     MB    224                 // Clear return value
      L     MW    220                 // Incontainer start address
      T     MW    200                 //
      L     3
      +I
      T     MW    220                 // Point to next index
      L     MW    222                 // Outcontainer start address
      T     MW    202
//
      L     MW    200                 // Get HIGH byte of index
      ITD
      SLD   3
      T     MD    242
      L     EB [MD 242]               // Byte ready..
      T     MW    206                 // Store HIGH in MW206
//
      L     MW    200
      L     1
      +I
      T     MW    200
      L     MW    200                 // Get MIDDLE byte of index
      ITD
      SLD   3
      T     MD    242
      L     EB [MD 242]               // Byte ready..
      T     MW    208
//
      L     MW    200
      L     1
      +I
      T     MW    200
      L     MW    200                 // Get LOW byte of index
      ITD
      SLD   3
      T     MD    242
      L     EB [MD 242]               // Byte ready...
      T     MW    210
//
      L     MB    207                 // Calculate nbr of bytes to read
      SRW   4
      L     W#16#7
      UW
      T     MB    212                 // This is the size from index
//
      U(
      ON    M     212.0
      ON    M     212.1
      )
      UN    M     212.2
      SPB   TR
//
      U     M     212.2
      SPB   WEI1
      L     4                         // Convert length to 4 bytes
      SPA   TR
WEI1: U(
      O     M     212.0
```

```
SIMATIC              ...ion 1\CPU315-2DP1\S7 Program(3)\AP-off\FC111  02/12/1997 12:57:52
      O     M    212.1
      )
      SPB   WEI2
      L     6                          // Convert length to 6 bytes
      SPA   TR
WEI2: U     M    212.1
      SPB   WEI3
      L     8                          // Convert length to 8 bytes
      SPA   TR
WEI3: U     M    212.0
      SPB   WEI4
      L     12                         // Convert length to 12 bytes
      SPA   TR
WEI4: L     16                         // Convert length to 16 bytes
TR:   T     MB   212                   // Save length in MB212
      NOP   0
```

Network: 2 READ MEMORY

This network handles the reading of device memory (MB/MW/MD/QB/QW/QD/IB/IW/ID)

```
      R     #IsX
      R     #IsY
      R     #IsM
      UN    M    207.0                 // Read memory bit?
      O     M    207.1
      O     M    207.2
      O     M    207.3
      SPB   TstY
      S     #IsM
      SPA   GOON
TstY: NOP   0
      UN    M    207.1                 // Read outputs ?
      ON    M    207.0
      O     M    207.2
      O     M    207.3
      SPB   TstX
      S     #IsY
      SPA   GOON
TstX: NOP   0
      UN    M    207.1                 // Read inputs ?
      O     M    207.0
      O     M    207.2
      O     M    207.3
      SPB   END2
      S     #IsX
GOON: NOP   0
//
      L     MW   202                   // Outcontainer pointer
      T     MW   216
//
      L     MB   209                   // Index middle in upper position
      SLW   8
      L     MB   211
      OW
      T     MW   210
//
      O     M    207.4
      O     M    207.5
      O     M    207.6
      SPB   M202                       // IF MORE THEN ONE BIT
//
//--------   HANDLING OF BIT READING (NOT USED IN E700 V1.1x)  ---------
//
      L     MW   210                   // BIT NUMBER
      L     W#16#7
      UW
      ITD
//
      L     MW   210                   // BYTE NUMBER
      SRW   3
      ITD
      SLW   3
      OD                               // BIT AND BYTE
      T     MD   242                   // POINTER TO MEMORY BIT
//
      L     MW   202                   //
      ITD
```

```
SIMATIC              ...ion 1\CPU315-2DP1\S7 Program(3)\AP-off\FC111  02/12/1997 12:57:52
      SLW   3
      T     MD   246
//
      UN    #IsM
      SPB   L1
      L     1
      U     M [MD 242]
      SPB   ACT2
      L     0
      SPA   ACT2
L1:   NOP   0
      UN    #IsX
      SPB   L2
      L     1
      U     E [MD 242]
      SPB   ACT2
      L     0
      SPA   ACT2
L2:   NOP   0
      L     1
      U     A [MD 242]
      SPB   ACT2
      L     0
      SPA   ACT2
ACT2: T     AB [MD 246]
      SPA   M204
//
//------  HANDLING OF BYTE READING (USED BY E700 V1.1x)  -----------
//
M202: NOP   0
      L     MW   210                    // Memory address from index
      SRW   3                           // Divide by 8
      T     MW   214
M206: NOP   0
      L     MW   214
      ITD                               // Make it double word
      SLW   3                           // Put byte in upper position (->bit =0)
      T     MD   242                    // This is the pointer to the data source
//
      L     MW   202                    // Output container pointer
      ITD                               // Make it double word
      SLW   3                           // Put byte in upper position
      T     MD   246                    // This is the pointer to the data destination
//
      UN    #IsM
      SPB   L10
      L     MB [MD 242]                 // Get byte from data source
      SPA   LDON
L10:  NOP   0
      UN    #IsX
      SPB   L11
      L     EB [MD 242]
      SPA   LDON
L11:  NOP   0
      L     AB [MD 242]
LDON: NOP   0
      T     AB [MD 246]                 // Store it in the output container
//
      L     MW   214                    // Increment data source pointer
      L     1
      +I
      T     MW   214
//
      L     MW   202                    // Increment data destination pointer
      L     1
      +I
      T     MW   202
//
      L     MW   216                    // Check if all bytes read according to data length in index
      -I
      L     MB   212
      <I
      SPB   M206                        // Go for next byte...
M204: NOP   0
      L     MW   202
      T     MW   222
END2: NOP   0
```

```
SIMATIC              ...ion 1\CPU315-2DP1\S7 Program(3)\AP-off\FC111 02/12/1997 12:57:52

Network: 3      READ DATABASE WORDS
This network handles the reading of database words (DBB/DBW/DBD)

      UN    M    207.3                  // Read database ?
      SPB   END3
//
      L     MW   202                    // Outcontainer pointer
      T     MW   216
//
//
      L     #DataBase                   // Number of the database to use
      T     MW   100
      AUF   DB [MW 100]                 // Open the database
LP3:  NOP   0
      L     MW   210                    // Database device address
      ITD
      SLD   3
      T     MD   242                    // This is the pointer to the data source
//
      L     MW   202                    // Output container pointer
      ITD                               // Make it double word
      SLW   3                           // Put byte in upper position
      T     MD   246                    // This is the pointer to the data destination
//
      L     DBB [MD 242]                // Get byte from data source
      T     AB [MD 246]                 // Store it in the output container
//
      L     MW   210                    // Increment data source pointer
      L     1
      +I
      T     MW   210
//
      L     MW   202                    // Increment data destination pointer
      L     1
      +I
      T     MW   202
//
      L     MW   216                    // Check if all bytes read according to data length in index
      -I
      L     MB   212                    // Nbr of bytes to read
      <I
      SPB   LP3
      L     MW   202
      T     MW   222
END3: NOP   0
```

SIMATIC ...ion 1\CPU315-2DP1\S7 Program(3)\AP-off\FC112 02/12/1997 12:57:58

FC112 - <Off-line>

Name: WRI_IND **Family:**
Author: Sia **Version:** 00.01
Time stamp Code: DT#1997-02-11-13:20:50.000
Interface: DT#1996-11-21-15:32:33.000
Length (Block / MC7 Code / Data): 00716 00572 00002

Address	Decl.	Symbol	Data Type	Initial Value	Comment
0.0	in	DataBase	WORD	W#16#0	
	out				
	in_out				
0.0	temp	IsM	BOOL		

This function handles one write index

Network: 1 INDEX HANDLING

```
      L     0
      T     MB   224                  // Clear return value
      L     MW   220                  // Incontainer start address
      T     MW   200
      L     3
      +I
      T     MW   220                  // Pointer to next index
//
      L     MW   200                  // Get HIGH byte of index
      ITD
      SLD   3
      T     MD   242
      L     EB [MD 242]               // Byte ready..
      T     MW   206                  // Store HIGH in MW206
//
      L     MW   200                  // Get MIDDLE byte of index
      L     1
      +I
      T     MW   200
      ITD
      SLD   3
      T     MD   242
      L     EB [MD 242]               // Byte ready..
      T     MW   208
//
      L     MW   200
      L     1
      +I
      T     MW   200                  // Get LOW byte of index
      ITD
      SLD   3
      T     MD   242
      L     EB [MD 242]               // Byte ready..
      T     MW   210
//
      L     MW   200
      L     1
      +I
      T     MW   200                  // Point to the data to be written
//
      L     MB   207                  // Calculate nbr of byte to write
      SRW   4
      L     W#16#7
      UW
      T     MB   212                  // Tis is the size from index
      U(
      UN    M    212.1
      UN    M    212.2
      )
      O
      U(
      UN    M    212.0
      U     M    212.1
      UN    M    212.2
      )
      SPB   ENDE
      U     M    212.2
      SPB   WEI1
      L     4                         // Convert length to 4 bytes
      SPA   TR
WEI1: U     M    212.1
      O     M    212.0
      SPB   WEI2
```

```
SIMATIC                ...ion 1\CPU315-2DP1\S7 Program(3)\AP-off\FC112  02/12/1997 12:57:58
      L     6                          // Convert length to 6 bytes
      SPA   TR
WEI2: U     M     212.1
      SPB   WEI3
      L     8                          // Convert length to 8 bytes
      SPA   TR
WEI3: U     M     212.0
      SPB   WEI4
      L     16                         // Convert length to 16 bytes
      SPA   TR
WEI4: L     32                         // Convert length to 32 bytes
TR:   T     MB    212
ENDE: NOP   0
```

Network: 2 WRITE MEMORY

This networks writes data to device memory (MB/MW/MD) and outputs (QB/QW/QD)

```
      UN    M     207.0                // Write M ?
      O     M     207.1
      O     M     207.2
      O     M     207.3
      SPB   TstY
      S     #IsM                       // Flag indicating M
      SPA   ST
TstY: NOP   0
      UN    M     207.0                // Write outputs ?
      ON    M     207.1
      O     M     207.2
      O     M     207.3
      SPB   END2
      R     #IsM
ST:   NOP   0
//
      L     MW    202                  // Outcontainer pointer
      T     MW    216
//
//
      O     M     212.0
      O     M     212.1
      O     M     212.2
      SPB   M202
      L     MW    210                  // Calculate bit number from device address
      L     W#16#FF
      UW
      L     MW    208
      SLW   8
      OW
      L     8
      /I
      T     MW    242
//
      L     MW    210                  // Calculate byte number from device address
      L     W#16#FF
      UW
      L     MW    208
      SLW   8
      OW
      L     8
      MOD
      T     MW    244
//
      L     MW    242                  // Pointer to bit
      ITD
      SLW   3
      L     MW    244
      ITD
      +D
      T     MD    242                  // Now, this is the pointer to the correct memory bit+byte
//
      L     MW    200                  //
      ITD
      SLD   3
      T     MD    246
//
      L     EB [MD 246]                // Get ON/OFF-status from input container
      T     MB    213
      UN    #IsM                       // Is it memory bit ?
      SPB   Is_Y
      U     M     213.0
      =     M [MD 242]                 // Write the data to the memory bit!
      SPA   AFTR
Is_Y: NOP   0
```

```
SIMATIC              ...ion 1\CPU315-2DP1\S7 Program(3)\AP-off\FC112 02/12/1997 12:57:58
      L     MB   213
      U     M    213.0
      =     A [MD 242]                  // Write the data to the output bit!
//
AFTR: NOP   0
      L     MW   210                    // Increase input container pointer
      L     1
      +I
      T     MW   210
//
      L     MW   202                    // Increase output container pointer
      L     1
      +I
      T     MW   202
//
      L     MW   202
      T     MW   222
      SPA   END2
//------  HANDLING OF BYTE READING (USED BY E700 V1.1x)  -----------
//
M202: NOP   0
      L     MW   210                    // Memory address from index
      SRW   3                           // Divide by 8
      T     MW   214
      L     MW   200
      T     MW   202
LOOP: NOP   0
      L     MW   214
      ITD                               // Make it double word
      SLW   3                           // Put byte in upper position (->bit =0)
      T     MD   242                    // This is the pointer to the data DESTINATION
//
      L     MW   202                    // Output container pointer
      ITD                               // Make it double word
      SLW   3                           // Put byte in upper position
      T     MD   246                    // This is the pointer to the data SOURCE
//
      L     EB [MD 246]                 // Get data from container
      UN    #IsM
      SPB   L10
      T     MB [MD 242]                 // Write byte to data destination
      SPA   LDON
L10:  NOP   0
      T     AB [MD 242]
LDON: NOP   0
//
      L     MW   214                    // Increment data source pointer
      L     1
      +I
      T     MW   214
//
      L     MW   202                    // Increment data destination pointer
      L     1
      +I
      T     MW   202
//
      L     MW   216                    // Check if all bytes read according to data length in index
      -I
      L     MB   212
      <I
      SPB   LOOP                        // Go for next byte...
      NOP   0
      L     MW   202
      T     MW   222
END2: NOP   0



Network: 3      Write of data word
      UN    M    207.3                  // Write dataword ?
      SPB   END3
      L     MW   200
      T     MW   216
//
//
      L     #DataBase
      T     MW   100
      AUF   DB [MW 100]                 // Open desired database
//
LP3:  NOP   0
      L     MW   200                    // Calculate addr in inp.conatiner to data to write
      ITD
      SLD   3
```

```
SIMATIC                ...ion 1\CPU315-2DP1\S7 Program(3)\AP-off\FC112  02/12/1997 12:57:58
      T     MD   242
//
      L     MW   210                  // Calculate position in database
      ITD
      SLD   3
      T     MD   246
//
      L     EB [MD 242]               // Get byte from input container
      T     DBB [MD 246]              // Store it in the database
//
      L     MW   210                  // Next input data byte
      L     1
      +I
      T     MW   210
//
      L     MW   200                  // Next destination address
      L     1
      +I
      T     MW   200
//
      L     MW   216                  // Check if more bytes to write
      -I
      L     MB   212
      <I
      SPB   LP3
      L     MW   200                  // Done!!
      T     MW   220
END3: NOP   0
      L     0
      L     MB   100
      ==I
      SPB   m1
      T     MB   172
m1:   NOP   0
```

Printed in Germany / 1997

MITSUBISHI ELECTRIC EUROPE
FACTORY AUTOMATION
GOTHAER STR. 8, D-40880 RATINGEN
TEL.: (0 21 02) 4 86-10, FAX: (0 21 02) 4 86-1 12