BAUMÜLLER

Option Board Multifunction Module MFM-01 for Ωmega Drive-Line II

Technical Description
and Operating Instructions
Edition: July 2001

E	5.00032.02

BAUMÜLLER

MFM-01 OPTION BOARD FOR ΩMEGA DRIVE-LINE II

Technical Description and Operating Instructions

Edition: July 2001

Document no. 5.00032.02

This operation manual is intended as a complement to the technical description and the operation manual of the apparatus.

BEFORE CARRYING OUT COMMISSIONING, CAREFULLY READ AND OBSERVE THE OPERATING INSTRUCTIONS AND SAFETY INFORMATION

This document contains all the information necessary to correctly use the products it describes. It is intended for specially trained, technically qualified personnel who are well-versed in all warnings and commissioning activities.

The equipment is manufactured using state-of-the-art technology and is safe in operation. It can safely be installed and commissioned and functions without problems if the safety information is followed.

You may not carry out commissioning until it has been established that the machine into which this component is to be installed complies with the specifications of the EC machine guidelines.

This technical description/these operating instructions invalidate all previous descriptions of the corresponding product. Within the scope of further development of our products, Baumüller GmbH reserves the right to change their technical data and handling.

Manufacturer and Supplier's Address: Baumüller Nürnberg GmbH
Ostendstr. 80
D-90482 Nürnberg
Tel. ++49 (0)9 11/54 32 - 0
Fax ++49 (0)9 11/54 32 - 1 30

Country of Origin: Germany

Date of Manufacture: Determined from the serial number on the equipment

TABLE OF CONTENTS

Table of Contents

1 SAFETY INFORMATION

General Information

These operating instructions contain all the information necessary for correct operation of the products described. The document is intended for specially trained, technically qualified personnel who are well-versed in all warnings and commissioning activities.

The equipment is manufactured using state-of-the-art technology and is safe in operation. It can safely be installed and commissioned and functions without problems if the safety information in these operating instructions is followed.

Danger Information

On the one hand, the information below is for your own personal safety and on the other to prevent damage to the described products or to other connected equipment.

In the context of the operating instructions and the information on the products themselves, the terms used have the following meanings:

DANGER

This means that **death, severe personal injury**, or **damage to property will** occur unless appropriate safety measures are taken.

WARNING

This means that **death, severe personal injury**, or damage to property **may** occur unless appropriate safety measures are taken.

NOTE

This draws your attention to **important information** about the product, handling of the product or to a particular section of the documentation.

Qualified Personnel

In the context of the safety-specific information in this document or on the products themselves, qualified personnel are considered to be persons who are familiar with setting up, assembling, commissioning and operating the product and who have qualifications appropriate to their activities:

- Trained or instructed or authorized to commission, ground and mark circuits and equipment in acordance with recognized safety standards.
- Trained or instructed in accordance with recognized safety standards in the care and use of appropriate safety equipment.

Appropriate Use

WARNING

You may only use the equipment/system for the purposes specified in the operating instructions and in conjunction with the third-party equipment and components recommended or authorized by BAUMÜLLER NÜRNBERG GmbH.

For safety reasons, you must not change or add components on/to the equipment/system.

The machine minder must report immediately any changes that occur which adversely affect the safety of the equipment/system.

2 TECHNICAL DATA

2.1 General

The multifunction module (MFM-01) is an option board for the Ωmega Drive-Line II.

Using this option board, you add to the Ωmega Drive-Line II:

- 8 digital inputs
- 8 digital outputs
- 4 analog inputs
- 2 analog outputs

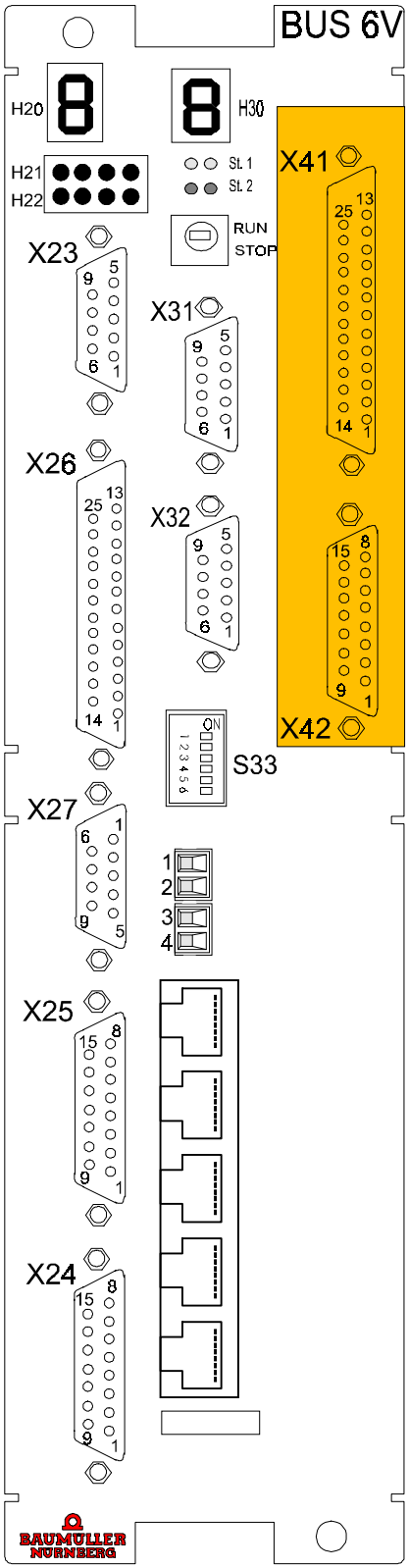
In addition, it is possible to trigger event tasks from the multifunction module by means of two different interrupt requests (IRQs) on the Ωmega Drive-Line II. The multifunction module offers closed A/D conversion as the interrupt source.

You can run the multifunction module in option slot 1 or 2 as desired.

2.2 Technical data of the multifunction module

Digital inputs:	
Quantity	8
Potential separation	Optocoupler
Nominal signal value	+24 V industrial logic
Signal level 1-signal	+15 ... +30 V
0-signal	0 ... +5 V
Input current	2.5 mA
Input delay	5 ms
Digital outputs (cannot be read-back, see “Structure Elements of MFM_WRITE_BMSTRUCT” on page 18)	
Quantity	8
Potential separation	Optocoupler
Nominal signal value	+24 V industrial logic
Signal level 1-signal	+18 ... +24 V
0-signal	0 ... +3 V
Output current	450 mA (sustained short circuit-proof)
Analog inputs (sampled at same time):	
Quantity	4 differential
Input voltage range	±10 V
A/D converter:	
Resolution	12-bit
Value of LSB	4.88 mV
Maximum linearity error	±½ LSB
Conversion time	34 µs
Analog outputs (cannot be read-back, see “Structure Elements of MFM_WRITE_BMSTRUCT” on page 18)	
Quantity	2
Output voltage range	± 10 V
Output current	5 mA
Output resistance	20 Ω
A/D converter:	
Resolution	12-bit
Value of LSB	4.88 mV
Linearity error	±1 LSB
Ambient conditions	
Operating temperature	0 ... 55° C
Storage temperature	-30 ... 70° C

Sample configuration 1:



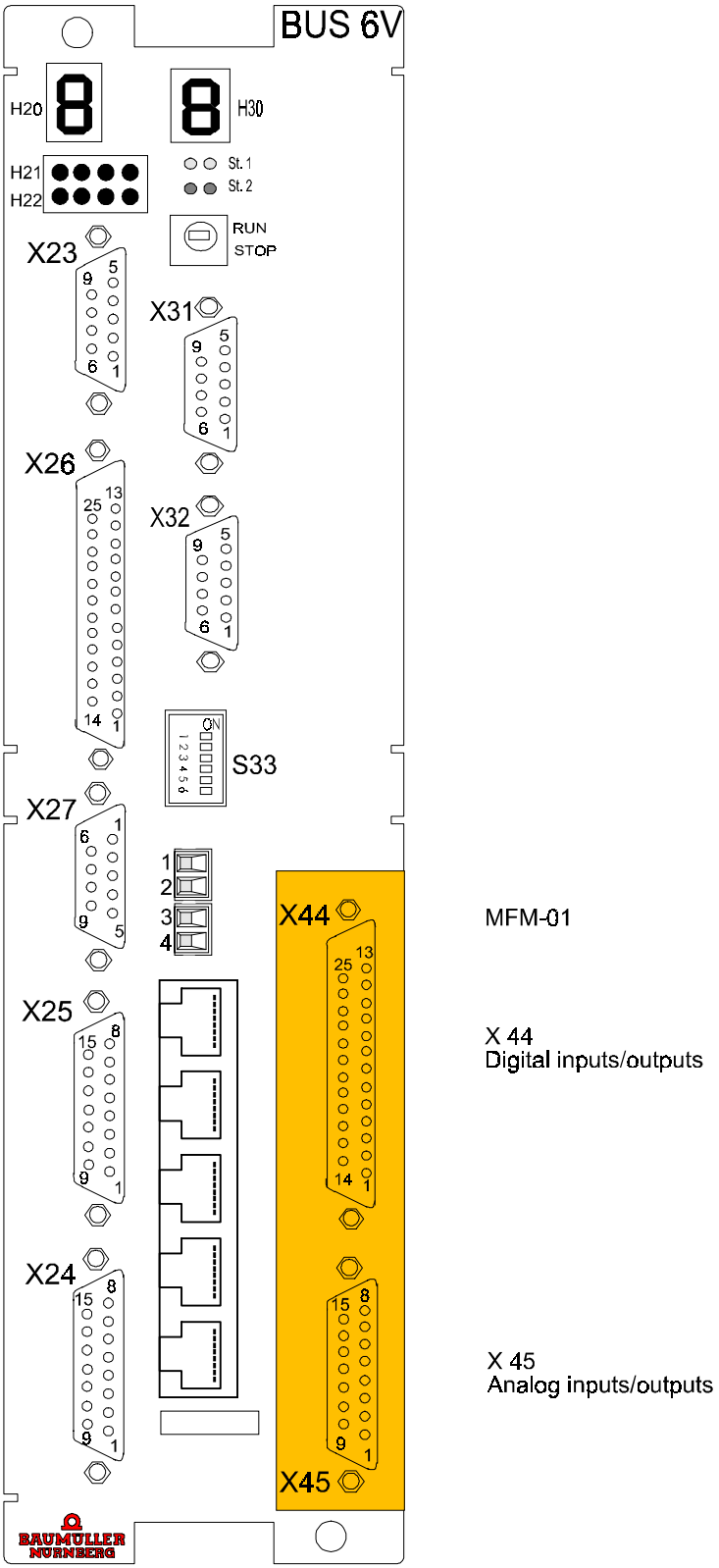
MFM-01

X 41
Digital inputs/outputs

X 42
Analog inputs/outputs

Ωmega Drive-Line II with MFM-01 in option slot 1

Sample configuration 2:



Ωmega Drive-Line II with MFM-01 in option slot 2

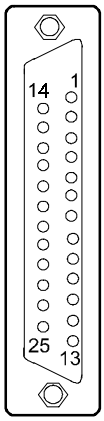
3 INSTALLATION

3.1 Pin Assignment

- Digital inputs/outputs

Connectors X 41 / X 44

Pin No.	Assignment
1	Digital output 1
2	Digital output 2
3	Digital output 3
4	Digital output 4
5	Digital output 5
6	Digital output 6
7	Digital output 7
8	Digital output 8
9	24 V ground for digital outputs
10	24 V ground for digital outputs
11	24 V ground for digital outputs
12	24 V ground for digital outputs
13	Short-circuit message for digital outputs
14	Digital input 1
15	Digital input 2
16	Digital input 3
17	Digital input 4
18	Digital input 5
19	Digital input 6
20	Digital input 7
21	Digital input 8
22	BSD (reference potential digital)
23	24 V external power supply for digital outputs
24	24 V external power supply for digital outputs
25	24 V external power supply for digital outputs

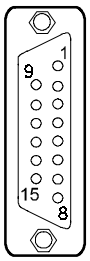


Digital outputs 1 to 8 need an external power supply of 24V (ground on pins 9 to 12 and +24 V on pins 23 to 25).

- Pin assignment

Connectors X 42 / X 45

Pin No.	Assignment
1	BSA (reference potential analog output)
2	Reserved
3	Analog output 2
4	Analog output 1
5	Analog input 4 (neg.)
6	Analog input 3 (neg.)
7	Analog input 2 (neg.)
8	Analog input 1 (neg.)
9	Reserved
10	BSA (reference potential analog output)
11	BSA (reference potential analog output)
12	Analog input 4 (pos.)
13	Analog input 3 (pos.)
14	Analog input 2 (pos.)
15	Analog input 1 (pos.)



NOTE

You must use screened cables with the cable screening being connected to the connector housings at both ends.

4 USING THE MFM-01 IN THE PROPROG WT II PROJECT

To be able to read the MFM-01's inputs or to set its outputs in the PROPROG wt II project, the program must access the registers of the MFM-01. To make these accesses easier, data types are declared that mimic the MFM-01's register structure (see “Data Types for MFM-01” on page 14).

The system uses these data types to declare variables that are assigned to the address of the option interface that is used (see “Declaring variables” on page 15). In this way, the variables mimic the MFM-01's registers (see “Structure Elements of MFM_READ_BMSTRUCT” on page 17 and see “Structure Elements of MFM_WRITE_BMSTRUCT” on page 18).

After this, it is possible to use the structure elements of the declared variables to access the MFM-01's register structure and to initialize the MFM-01 in this way (see “Initialization” on page 20) and to read and set respectively the inputs and outputs (see “Example” on page 22).

4.1 Data Types for MFM-01

Two structures are available for carrying out communication between the Ωmega Drive-Line II and the multifunction module. These structures are defined from library "BM_TYPES_20bd00" or above. Once you have integrated library "BM_TYPES_20bd00" or above in the project, you have available data types MFM_READ_BMSTRUCT and MFM_WRITE_BMSTRUCT.

Definition:

The MFM_READ_BMSTRUCT structure is defined as follows:

```
TYPE
   MFM_READ_BMSTRUCT:   STRUCT
      i_ANALOG_IN0  :   INT;
      i_ANALOG_IN1  :   INT;
      i_ANALOG_IN2  :   INT;
      i_ANALOG_IN3  :   INT;
      i_DUMMY0      :   INT;
      i_DUMMY1      :   INT;
      i_DUMMY2      :   INT;
      b_RESERVED    :   BYTE;
      b_DIGITAL_IN  :   BYTE;
      i_DUMMY3      :   INT;
   END_STRUCT;
```

The MFM_WRITE_BMSTRUCT structure is defined as follows:

```
   MFM_WRITE_BMSTRUCT:  STRUCT
      i_ANALOG_IN_START:INT;
      i_DUMMY0      :   INT;
      i_DUMMY1      :   INT;
      i_DUMMY2      :   INT;
      i_ANALOG_OUT0 :   INT;
      i_ANALOG_OUT1 :   INT;
      w_INIT_REGISTER:  WORD;
      i_DUMMY3      :   INT;
      b_DIGITAL_OUT :   BYTE;
      b_DUMMY0      :   BYTE;
   END_STRUCT;
END_TYPE
```

4.2 Declaring variables

You declare two global variables of types MFM_READ_BMSTRUCT and MFM_WRITE_BMSTRUCT. Using these variables and their structure elements (see “Structure Elements of MFM_WRITE_BMSTRUCT” on page 18 and see “Initialization” on page 20), it is possible to access the multifunction module.

NOTE

It is only possible to access the MFM-01's registers word-by-word. This means that it is only possible to use the structure elements of these variables to access the MFM-01.

In the PROPROG wt II project, you create a global variable of data type

`MFM_READ_BMSTRUCT`

and assign it to the base address of option interface 1

`%MW3.1000000`

Example:

```
_MFM_READ AT %MB3.1000000 : MFM_READ_BMSTRUCT;
```

Where:

`_MFM_READ`	is the variable name with the data type short designation "_" for STRUCT
`MFM_READ_BMSTRUCT`	is the variable's data type
`%MB3.1000000`	is the base address of option interface 1

Example of accessing an element of the structure:

```
_MFM_READ.b_DIGITAL_IN
```

Where:

_MFM_READ	is the variable name
b_DIGITAL_IN	is the element of the structure with the data type short designation "b" for BYTE

NOTE

In the following description, the variable name is replaced by an asterisk (*).

Additionally, in the PROPROGwtII project, you create a global variable of data type

`MFM_WRITE_BMSTRUCT`

and assign it to the base address of option interface 1

`%MW3.1000000`

Example:

```
_MFM_WRITE AT %MB3.1000000 : MFM_WRITE_BMSTRUCT;
```

Where:

`_MFM_WRITE`	is the variable name with the data type short designation "_" for STRUCT
`MFM_WRITE_BMSTRUCT`	is the variable's data type
`%MB3.1000000`	is the base address of option interface 1

Example of accessing an element of the structure:

`_MFM_WRITE.w_INIT_REGISTER`

Where:

`_MFM_WRITE`	is the variable name
`w_INIT_REGISTER`	is the element of the structure with the data type short designation "w" for WORD

NOTE

In the following description, the variable name is replaced by an asterisk (*).

You must assign both variables to the address of the option interface that is being used. The address of the option interface results from the option slot that you are using.

- Option slot 1 → Option interface 1 → address AT %MW3.1000000
- Option slot 2 → Option interface 2 → address AT %MW3.2000000

4.3 Structure Elements of MFM_READ_BMSTRUCT

Address mapping

Address	Data type	Designation
n + 0	INT	*.i_ANALOG_IN0
n + 2	INT	*.i_ANALOG_IN1
n + 4	INT	*.i_ANALOG_IN2
n + 6	INT	*.i_ANALOG_IN3
n + 15	BYTE	*.b_DIGITAL_IN

n = %MW3.1000000 on option interface 1
%MW3.2000000 on option interface 2

*) Corresponds, for example, to variable _MFM_READ (see “Declaring variables” on page 15).

- *.i_ANALOG_IN0
  Actual value of analog input 0.

+10V	16#7FF	2047
0 V	16#000	0
-10V	16#800	-2048

- *.i_ANALOG_IN1
  Actual value of analog input 1.

+10V	16#7FF	2047
0 V	16#000	0
-10V	16#800	-2048

- *.i_ANALOG_IN2
  Actual value of analog input 2.

+10V	16#7FF	2047
0 V	16#000	0
-10V	16#800	-2048

- *.i_ANALOG_IN3
  Actual value of analog input 3.

+10V	16#7FF	2047
0 V	16#000	0
-10V	16#800	-2048

- *.b_DIGITAL_IN
  Digital inputs bit 0 to bit 7 correspond to bits 0 to 7 in *.b_DIGITAL_IN.

4.4 Structure Elements of MFM_WRITE_BMSTRUCT

Address mapping

Address	Data type	Designation
n + 0	INT	*.i_ANALOG_IN_START
n + 8	INT	*.i_ANALOG_OUT0
n + 10	INT	*.i_ANALOG_OUT1
n + 12	WORD	*.w_INIT_REGISTER
n + 16	BYTE	*.b_DIGITAL_OUT

n = %MW3.1000000 on option interface 1
%MW3.2000000 on option interface 2

*) Corresponds, for example, to variable _MFM_WRITE (see “Declaring variables” on page 15).

NOTE

The MFM-01 registers that are used for MFM_WRITE_BMSTRUCT allow read accesses only. Write accesses to this address or this structure element are not possible. The system cannot display correctly the values of these registers or structure elements even in online mode.

- *.i_ANALOG_IN_START
  Register for starting A/D conversion with bit 14 set in *.w_INIT_REGISTER. Writing to this variable starts A/D conversion. The written datum is not relevant.

- *.i_ANALOG_OUT0
  Reference value of analog output 0

+10V	16#7FF	2047
0 V	16#000	0
-10V	16#800	-2048

- *.i_ANALOG_OUT1
  Reference value of analog output 1

+10V	16#7FF	2047
0 V	16#000	0
-10V	16#800	-2048

- *.w_INIT_REGISTER
  This register is for initializing the two interrupt request signals, and for choosing the start signal for A/D conversion.

  This register must be initialized, since its status is not defined after activation
  (see “Initialization” on page 20).

- *.b_DIGITAL_OUT
  Digital outputs 0 to 7 correspond to bits 0 to 7. The output byte of the digital outputs should be set to 0 at the transition from Run → Stop so that the outputs are Low.

4.5 Initialization

In initialization of the multifunction module, it is only necessary to write to *.w_INIT_REGISTER (see “Structure Elements of MFM_WRITE_BMSTRUCT” on page 18). This is possible at booting or resetting.

* Corresponds, for example, to variable _MFM_WRITE.

4.5.1 Configuring the Interrupt Requests

You can use two different interrupt request (IRQ1 and IRQ2) to trigger event tasks on the Ωmega Drive-Line II. On the multifunction module, the end of A/D conversion is available as an event for triggering an interrupt request:

Bits 8 to 11 of initialization register *.w_INIT_REGISTER (see “Structure Elements of MFM_WRITE_BMSTRUCT” on page 18) are used for initializing the multifunction module interrupt request.

* Corresponds, for example, to variable _MFM_WRITE.

Bits 8 and 9 in the initialization register are used to assign to interrupt request 1 and 2 (IRQ1 and IRQ2) the end of A/D conversion as an event.

Bit 8	Bit 9	Option 1	Option 2
0	0	IRQ1	IRQ2
0	1	IRQ2	IRQ1

Bits 10 and 11 in the initialization register are used to enable IRQ1 and IRQ2. Only the enabled IRQs can trigger event tasks on the Ωmega Drive-Line II.

Bit 10	IRQ1 on option 1	IRQ2 on option 2
0	Disable	Disable
1	Enable	Enable

Bit 11	IRQ2 on option 1	IRQ1 on option 2
0	Disable	Disable
1	Enable	Enable

You use function block (FB) "INTR_SET" from firmware library "SYSTEM2_DLII_20bd00" or above to initialize the event tasks on the Ωmega Drive-Line II. Using this FB, you can assign the individual IRQs of the multifunction module to event tasks and enable them.

4.5.2 Configuring the Start Signal for A/D Conversion

The system must start simultaneous conversion of the four analog inputs to digital values by a start signal. There are two options for generating this signal.

1. Cyclical A/D conversion

   Conversion is triggered by Trigger 2. This is an option board trigger for the multifunction module. This trigger is generated by Ωmega Drive-Line II and is not dependent on the option interface (option slot 1 or option slot 2) that you are using.

   This trigger is configured using function block "OPT_INIT" from library "SYSTEM1_DLII_20bd00" or above. You can assign different trigger sources to Trigger 2. It is possible to use different signals from the Ωmega Drive-Line II as trigger sources (see the Ωmega Drive-Line II technical description).

2. A/D conversion after write access

   The system starts A/D conversion after writing "*.i_ANALOG_IN_START" (see “Structure Elements of MFM_WRITE_BMSTRUCT” on page 18). The written datum is not relevant.

Bit 14 in the initialization register "*.w_INIT_REGISTER" (see “Structure Elements of MFM_WRITE_BMSTRUCT” on page 18) specifies the type of start signal for A/D conversion.

Bit 14	A/D conversion
0	Cyclical
1	After write access

Bit 15 in the initialization register is irrelevant

* Corresponds, for example, to variable _MFM_WRITE.

4.6 Example

Multifunction module on option interface 1 (option slot 1)

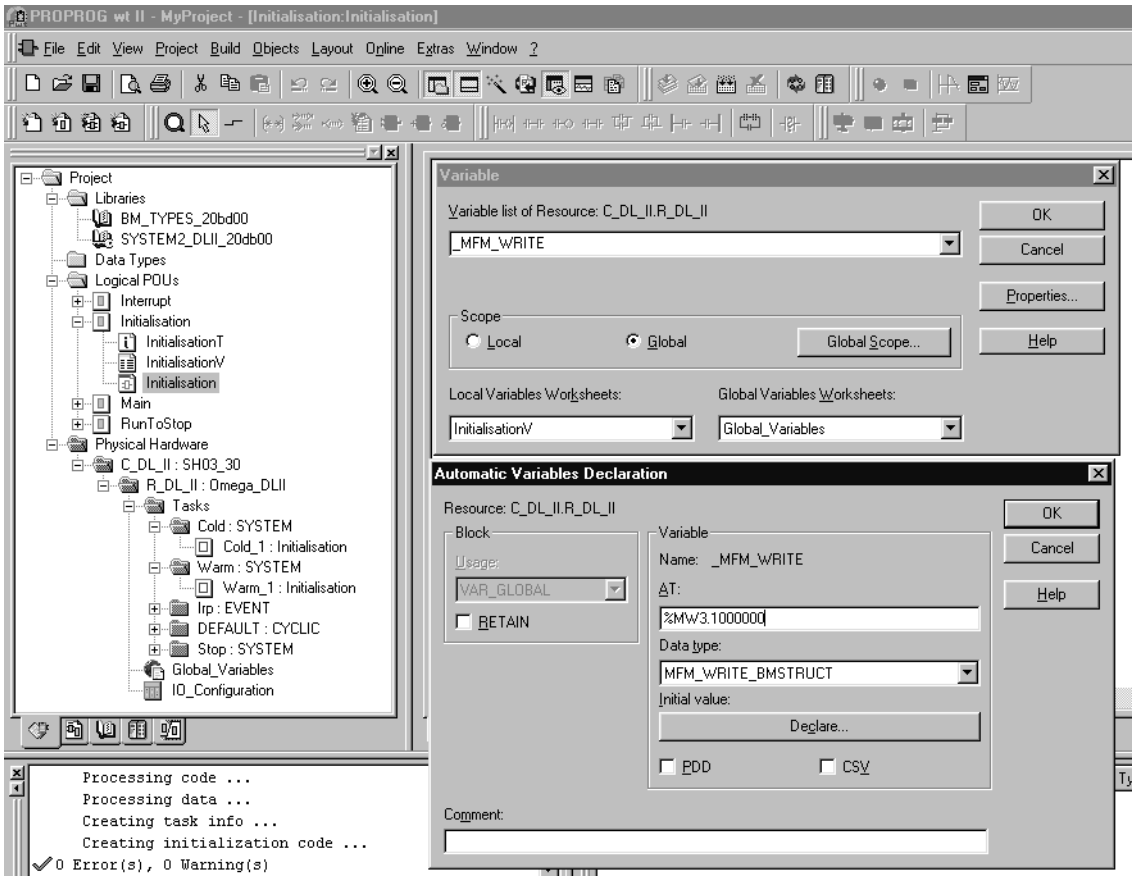
In a CANsync event task, you want to start analog/digital conversion by means of a write access.

The system reads the analog input values after process data communication.

4.6.1 Initialization

– Create a POU (Program Organization Unit), see PROPROG wt II Manual) of type program for initialization

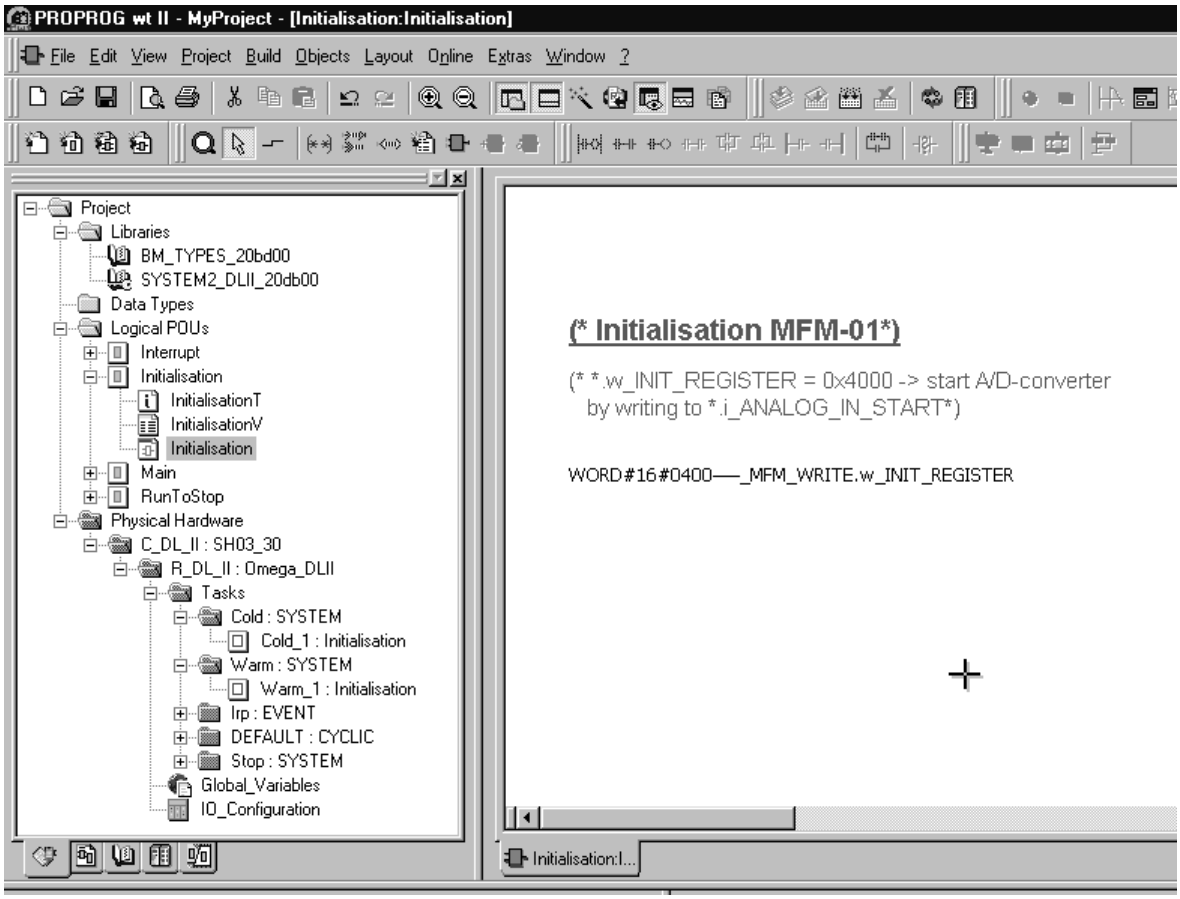
– Declare variables for writing to the MFM-01



– Write 0x0400 (Bit 14 = TRUE: A/D conversion after write access) to the MFM-01's initialization register

– Set up the system tasks for booting and resetting

– The POU for the initialization program is assigned to these tasks

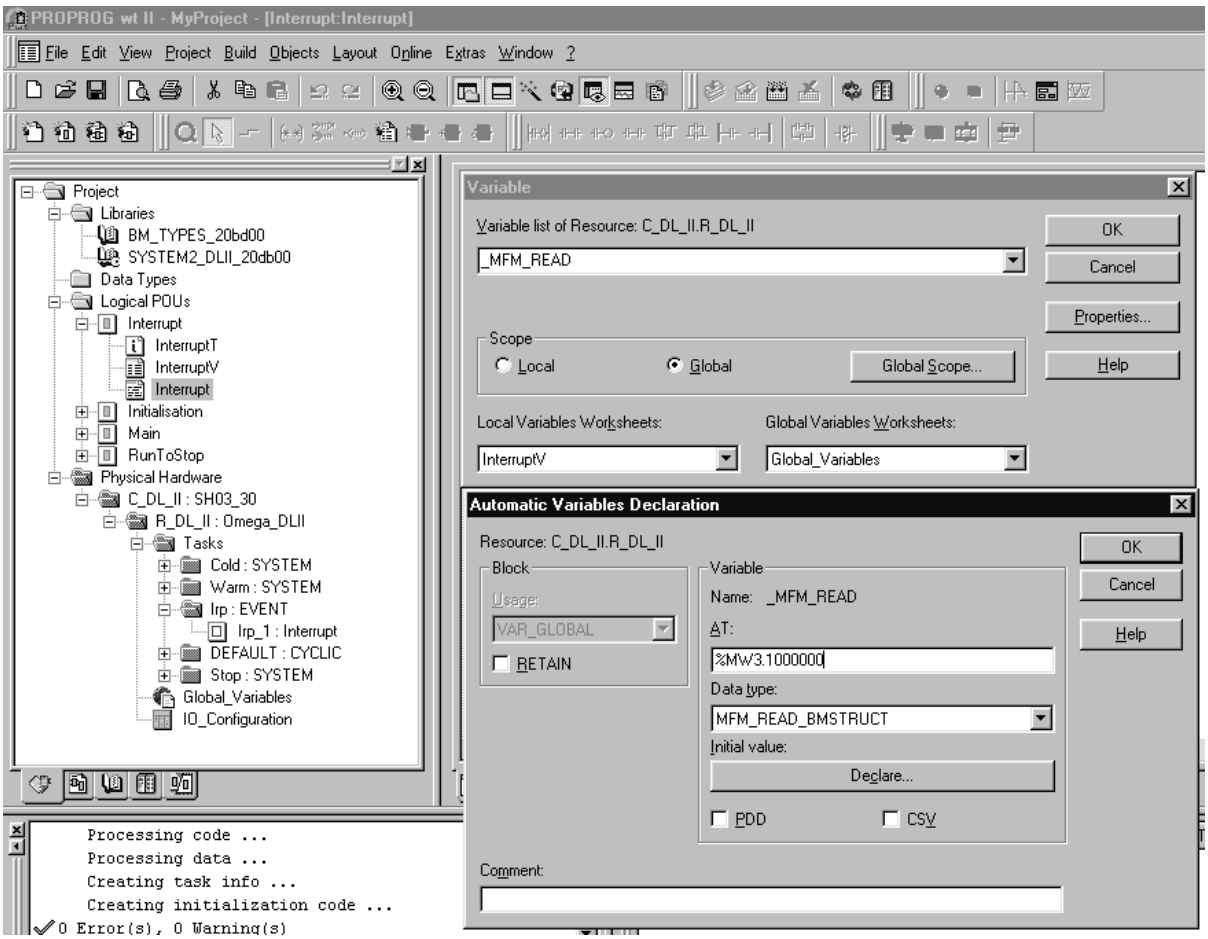

PROPROG wt II - MyProject - [Initialisation:Initialisation]
File Edit View Project Build Objects Layout Online Extras Window ?
Project
Libraries
BM_TYPES_20bd00
SYSTEM2_DLII_20db00
Data Types
Logical POUs
Interrupt
Initialisation
InitialisationT
InitialisationV
Initialisation
Main
RunToStop
Physical Hardware
C_DL_II : SH03_30
R_DL_II : Omega_DLII
Tasks
Cold : SYSTEM
Cold_1 : Initialisation
Warm : SYSTEM
Warm_1 : Initialisation
Irp : EVENT
DEFAULT : CYCLIC
Stop : SYSTEM
Global_Variables
IO_Configuration
(* Initialisation MFM-01*)
(* *.w_INIT_REGISTER = 0x4000 -> start A/D-converter
by writing to *.i_ANALOG_IN_START*)
WORD#16#0400——_MFM_WRITE.w_INIT_REGISTER
Initialisation:I...

4.6.2 Event Task

– Create a POU of type program for the event task

– Program the event task program

1. Declare variables for reading the MFM-01 with assignment of the address of the option interface that is being used

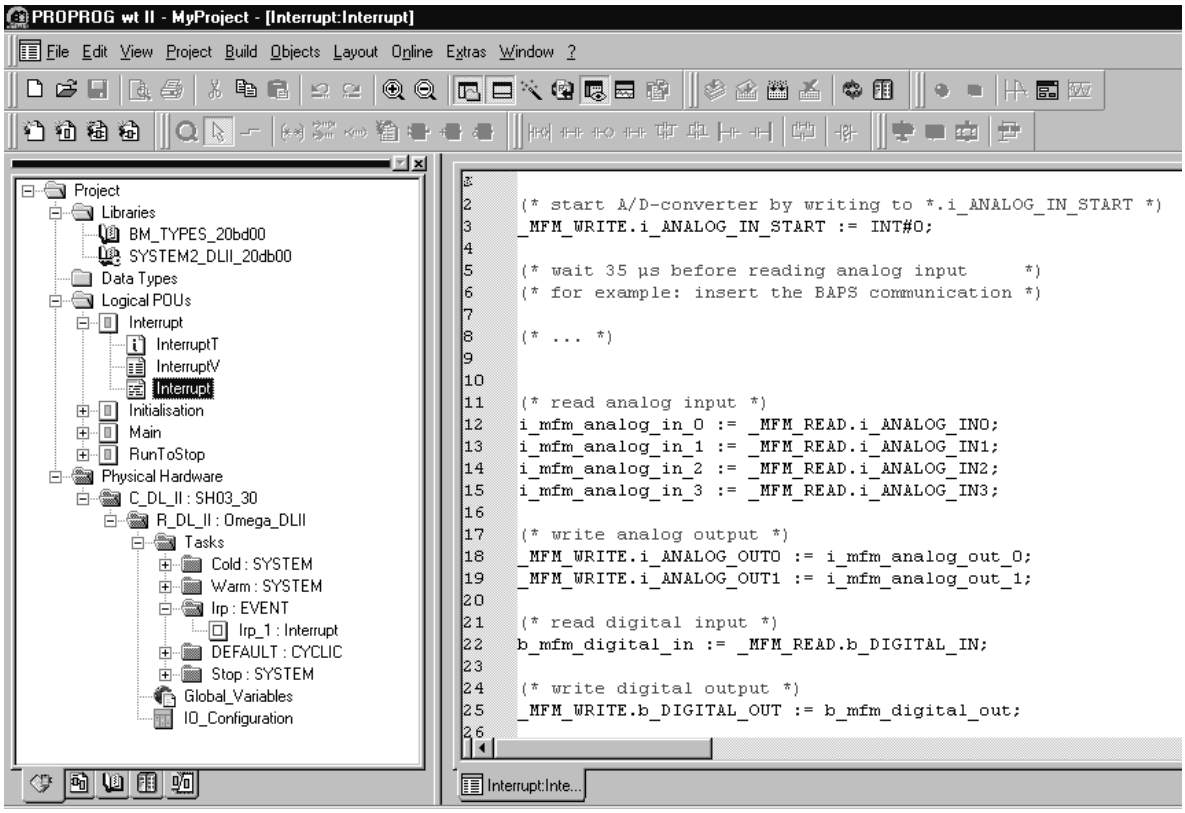


2. The variable for writing to the MFM-01 was already declared at initialization and you can use it here too.

3. Write to *.i_ANALOG_IN_START to start A/D conversion.

4. Read and write the MFM-01's inputs and outputs.

    * Corresponds, for example, to variable _MFM_WRITE



– Set up the CANsync event task

  Properties

    Task type: Event

    Type: Task

  Settings:

    Event: SYNC signal network (CANsync)

    Bypass: Yes

  These settings result from the selected interrupt source (see Ωmega Drive-Line II technical description)

– The POU for the event task program is assigned to this event task

5 INDEX