SIEMENS

Cerberus® MM7000
Cerberus® MF7000
MUX/DEMUX System

PLC Functions

Fire & Security Products

Siemens Building Technologies Group

Data and design subject to change without notice. / Supply subject to availability.
© Copyright by
Siemens Building Technologies AG

We reserve all rights in this document and in the subject thereof. By acceptance of the document the recipient acknowledges these rights and undertakes not to publish the document nor the subject thereof in full or in part, nor to make them available to any third party without our prior express written authorization, nor to use it for any purpose other than for which it was delivered to him.

1 Preface

The MF7033 and MM7033 belong to the equipment family DMS7000 and the common base of this family is the MPU module E2H081.
The MF7033 is used to instigate predefined interventions on the occurence of specified events. These events can be telegrams received via the communication network or lines activated on a MUX input. They are handled internally by logic or timers, and lamps or relays can be activated or telegrams transmitted to other members of the system. The MM7033 can also make the same connections as the MF7033 using logic or timers etc, but the area of application is different. The MF7033 is defined as evaluation equipment whereas the MM7033 is situated in the acquisition level.

2 Conventions

The folloing terminology is used throughout this manual:
PLC file is the MF7033 data file (<sitename>.Q00).
D_IN is the element name for a MUX line.
D_OUT is the element name for a DMUX line.

3 Limitations, System Characteristics

3.1 Hardware

Mix module E2A050 is NOT allowed in MF7033!
Max. hardware : **12** Periphery modules E2A041 (MUX) or E2A032 (DMUX).
. 5 relay cards connected to PIA ports on PIA module E1H040
. (or directly on E2H081).
. See description below for PIA port assignment.

Allowed configuration of the PIA ports:

Port name in the PLC file	Port name with hardware E2H080/E1H040	Port name with hardware E2H081/E1H040
PIA1	P1 on E1H040=S	P1 on E2H081=S
PIA2	P2 on E1H040=xx	P2 on E2H081=xx
PIA3	P3 on E1H040=xx	P1 on E1H040=xx
PIA4	P4 on E1H040=xL	P2 on E1H040=xx
PIA5		P3 on E1H040=xx
PIA6		P4 on E1H040=xL

S=Power supply monitoring/
xL = 'Output' or 'Input' or 'LampTest' or 'Not in use.
xx = 'Output' or 'Input' or 'Not in use'.
NB : Only P4 on E1H040 is allowed for lamp test!

3.2 Software

It is not possible to specify a rigid figure for the maximum number of elements that can be defined in the digital PLC machine (MF/MM), as the limitation varies from one application to another.
Rules for calculating the maximum number of elements allowed in a particular application are described below.

3.2.1 Rules for Loadfactor Calculation

For an MF7033, a total of 20'000 loadfactors is available. For an integrated MF (MF7013/MF7023), the upper limit is 0'000.

			Basis load = peak
F_EQUAL 3 + n			(n = number of outputs)
F_OR 3 + n			(n = number of inputs)
F_LTH 5 +2n			(n = number of inputs)
F_AND 3 + n			(n = number of inputs)
F_NOT 3			
F_EXOR 5			
F_LATCH 8			
F_WAIT 7			
F_DELAY 8			
F_COUNT 7			
F_TOUT 9			
F_TIM 8			
F_DATE 8			
F_TSW 9			
Basis load			**Peak loadfactor by event**
MF_TGIx	10		20 + max. 200 by intersecting ranges
MF_TGI2p		15	20 + max. 200 by intersecting ranges
MF_CAKx		20	40 + max. 200 by intersecting ranges
MF_TGIO	100		
MF_ONLINE		10	Max. 128 is reasonable
MF_RTC	7		Max. 128 is reasonable
MF_TGOx		10	20 + max. 1000 by simultaneous activation
MM_xx		10	20 + max. 1000 by simultaneous activation

If the rules described above are followed, a maximum reaction time of 1 second is guaranteed from the activation of a switch or receipt of a telegram.

Symptoms if these rules are not followed:

1. Temporary transgression of the maximum loadfactor limit (processing events) may cause telegrams to queue up in heavy telegram traffic. This queue will disappear the moment the maximum loadfactor is no longer exceeded.
Consequences: Time functions will no longer be accurate (F_WAIT, F_TIM...).

2. Continuous transgression of the maximum loadfactor limit (basis load > 20'000 loadfactors) will cause the telegram queue to build up continually and never disappear.
Consequences: Responce times > 1 second (with rising tendency).

Timing tests are required if the basis load is more than 10% higher than the maximum permissible load factor.

How to carry out a timing test

1. Define PIA port P2 as an output PIA in the system data (sitename.A10) - P2 on E2H081 or P2 on E1H040/E2H080.
2. Connect test LEDs to P2 on the PIA module.
3. Connect a monitor to the service interface (see chapter: Using the service interface).
4. Activate the test by selecting 'Z' from the menu on the service interface, and set 'DebugFlag' to '1'.
5. Evaluation:
   LED-0 and LED-1 are flashing well.
   ➜ no performance problems.
   LED-0 is almost not flashing, but LED-1 is still flashing.
   ➜ MF is fully loaded.
   Almost no flashing visible on either LED
   ➜ performance problems
   ➜ contact development department, more accurate measurements are required.

4 Using the service interface

The MPU module E2H081 has a service interface. This interface is particularly useful when booting the equipment.What can be seen on the service interface?

- booting information
- missing files
- syntax errors in the PLC file are reported.
- system crash

Communication parameters:

- 9600 Baud
- 8 bit
- 1 stop bit
- NO parity

Use the CERBERUS modem.
VT100 mode is emulated.

5 Process Image

Using the PLC file as a basis, the process image is built up internally in the MF7033. Consequently the use of wildcards is limited; however a powerful new concept has been introduced - **DATA RANGE**. This allows ranges 'from start address - to end address' to be defined.

Example:
Assuming we want to evaluate all alarms from automatic detectors in zones 31 .. 47 of the control unit CZ10 (location number 111).
The parameters look like this:

```
Sector          :A    Location : 111
ADF1 Min        :A1   ADF2 Min : 31
ADF1 Max        :A1   ADF2 Max : 47
Data A          :85   Data B   : 01
```

A range can also be specified for several sections. The range:

```
ADF1 Min        : E1  ADF2           Min           :           31
ADF1 Max        : E5  ADF2 Max : 47
```

evaluates the zones from 31 .. 47 in the sections E1 .. E5.
It is easy to see that this gives a much better overview of the data, and there is never any doubt about which data is included in the evaluation and which is not.
Important for the address range in basic sector: From the telegram table (AH0.1) some address fields are described to be `Adf1 = 00 and Adf2 = 00`. To evaluate this type of telegrams, the address fields may not be defined `'00, 00, 00, 00'` (`Adf1Min, Adf2Min, Adf1Max, Adf2Max`) because the MF allocates `00 00` to internal values from `00` up to `0F` for aquisition equipment and data concentrator (sector F), and from `00` to `3F` for evaluation equipment.
Thus the correct entry for theese types of telegrams are:

	Adf1Min	Adf2Min	Adf1Max	Adf2Max
Aquisition equipment	00	00	00	**0F**
Data concentrator		00	00	**0F**
Evaluation equipment	00	00	00	**3F**

6 System data (.CFG, .SYS, .A10)

The configuration file (.CFG) specifies the name of the equipment program, how many drives (floppy or hard disk), and what kind of drives are present.
The boot file (.SYS) specifies the site name, language and paths for the different files.
The system data file (.A10) defines the major differences between the MF and the MM, but also values common to both MF and MM - such as location number, hardware configuration (number of MUX/DMUX modules) etc.

➜ See APPENDIX F ‘description of the system data files..' for a complete description of the system data files.

7 File Contents of a Site Disk

The following files must be present on a site disk.

7.1 MF7033 / MM7033

AUTOBOOT.SYS	Booting		file
DMS7000.CFG	Configuration		file
CF7000xx.CMD	Equipment		program
DF7000.LXC	Telegram		table
ERRORS.TXT	Error		list
<siteName>.A10	System	data	file
<siteName>.Q00	PLC-file		

7.2 MC7003 / MC7033

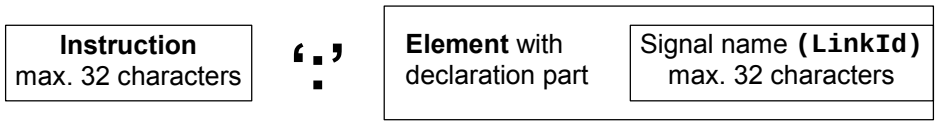
DF7000.LXC	Telegram		table
ERRORS.TXT	Error		list
<siteName>.A10	System	data	file
<siteName>.Q00	PLC-file		

8 Syntax of the MF7033/MM7033 PLC-File (<sitename>.Q00)

We strongly recommend that the structure of the PLC-file decribed below be used. If it is not used, the PLC machine may NOT work correctly in cases with internal feed back (LOOPS, logic before input elements etc.).
The typical structure of the PLC-file is shown below:
(element syntax in this example is not 100% correct, as a pseudocode has been used for simplicity)

```
HEADER
   Project...
   Purpose...
   ..... (anything but END)....
END

SEGMENT Digital_Inputs
   :D_IN(1:M_DI_1_1_S...
   :D_IN(1:M_DI_x_y_z...
END Digital_Inputs

SEGMENT Pre_Logic_FL
   :F_AND...
   :F_OR...
   :F_COUNT....
END Pre_Logic_FL

SEGMENT MM_Functions
   :MM_IA1...
   :MM_IC1...
END MM_Functions

SEGMENT MF_Functions_FL
   :MF_TGI2
   :MF_TGI3
END MF_Functions_FL

SEGMENT Post_Logic_FL
   :F_AND...
   :F_OR...
   :F_COUNT....
END Post_Logic_FL

SEGMENT Digital_Outputs
   :D_OUT(1:M_DO_1_1_S...
   :D_OUT(1:M_DO_x_y_z...
END Digital_Outputs
```

Definition of names:
Names after 'SEGMENT' and 'END' are SegmentNames. A segment name is an 'Identifier' (see definition of identifier below). The elements are put between the start and the end of a segment. Elements consist of an instruction followed by a ':' and then the element decription.



'**Text**: AND', '**text**:OR', '**text**:TGIx' . are 'Instructions' (see 'Instructions' below).

Identifiers:
Identifiers have a maximum length of 32 characters. They are 'case' sensitive. Allowed characters: 'A'..'Z', 'a'..'z', '0'..'9', '_'.

Instructions:
All elements (functions) start with a name. This name is optional and serves as a comment. We recommend that the instruction be used to give each element a meaningful name. As shown in the recommended PLC-file structure, the input section of a block is not in the same segment as the logic section of the same block. For clarity, it is usually convenient to give elements belonging to the same block the same name. This makes a PLC-file much easier to handle during later modifications.
The maximun length of the name is the same as for identifiers.

Comments:
A comment can be inserted at any time in the PLC-file. It can be recognized by '(*' as start of comment, and '*)' as end of comment. E.g. (* comment *)

InSignal:
Is an 'Identifer' (see explanation above), or a 'GlobalInSignalIdentifier':

M_DI_1_1_S..M_DI_16_48_S,	= actual state of digital input
M_DI_1_1_P..M_DI_16_48_P,	= occurence of positive edge
M_DI_1_1_N..M_DI_16_48_N,	= occurence of negative edge
M_PIAI_1_1_S..M_PIAI_10_8_S,	= actual state of PIA input
M_PIAI_1_1_P..M_PIAI_10_8_P,	= occurence of positive edge
M_PIAI_1_1_N..M_PIAI_10_8_N,	= occurence of negative edge

OutSignal:
Is an 'Identifer' (see explanation above), or a 'GlobalOutSignalIdentifier_S':

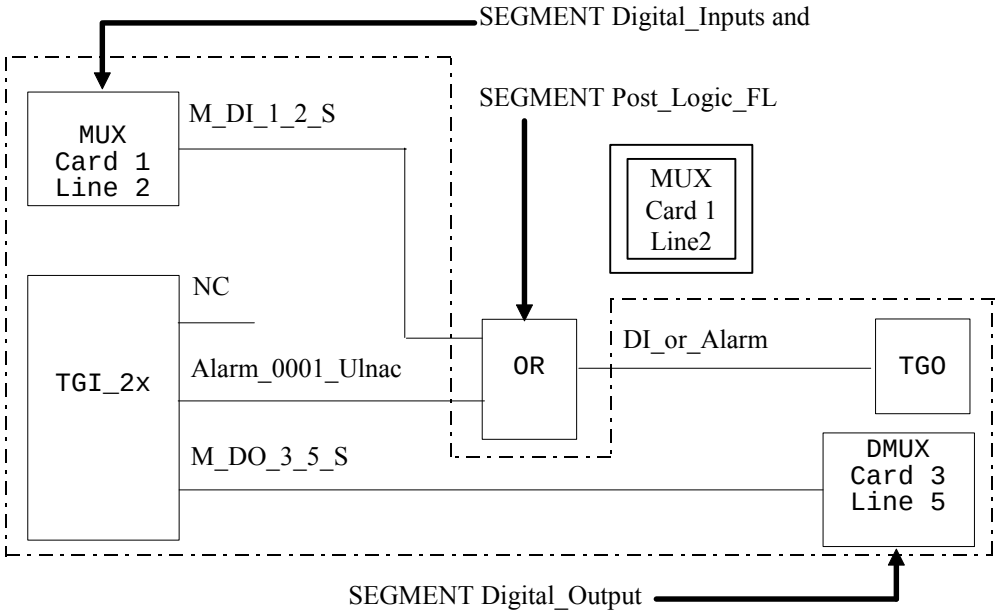
M_DO_1_1_S..M_DO_16_48_S,	= actual state of digital output
M_PIAO_1_1_S..M_PIAO_10_8_S	= actual state of PIA output
'GlobalOutSignalIdentifer_B':	
M_DO_1_1_B..M_DO_16_48_B,	= attribute flashing
M_PIAO_1_1_B..M_PIAO_10_8_B,	= attribute flashing

LinkId:
All signals not being GlobalInSignalIdentifiers (MUX inputs) or GlobalOutSignalIdentifiers (DMUX outputs) have invented names. The LinkId describes the 'soft wireing' of the PLC.

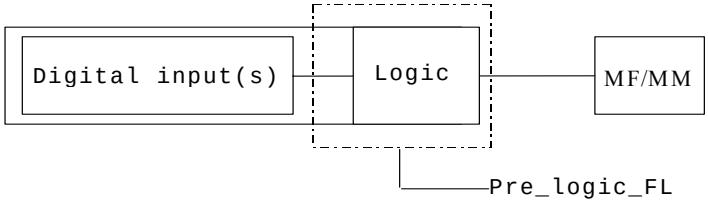
The maximum number of LinkId, not being
`GlobalIn(Out)SignallIdentifiers are 10000.`

All unused outputs of the elements must be declaired as not connected (**NC)**

Example of element connection with LinkId and usage of the segments:



The figure above shows how the different elements are connected. It also shows in what segment that the elements belong. Though one question is not answered. What do I do with the segment 'Pre_Logic_FL'?
A short description of the PLC machine is required to get an idea of the segment philosophy. (Allmost) all the connections and elements described in the PLC file are scanned within one cycle time (250ms).
The output of logic elements are set according to the values set on the input of the same element. The scanning starts with the first element defined in the PLC file, and stops with the last one defined in this file. Most MF elements are dependent of the process image, so that the outputs of these elements are set according to the values in the process image and the counters defined. Most MM elements react directly on the MUX inputs defined plus telegrams received. All these events must be treated within one cycle time. Digital inputs connected to a block of logic and then to MF or MM elements must be put in the segment Pre_Logic_FL. This is because the output of the logic must be defined before the input of the MF/MM elements, connected to the output of the logic, are scanned.



➜ See APPENDIX 'Syntax For The PLC files' for a complete syntax description.

9 Overview of all MF/MM elements

The table below shows all available elements, and in which equipment type each element is usable.

Input elements	Equipment type		Logic elements	Equipment type		Output-elements	Equipment type	
	MF	MM		MF	MM		MF	MM
D_IN	x	x	F_EQUAL	x	x	D_OUT	x	x
F_TIM	x	x	F_NOT	x	x	MF_TGO2	x	u
F_TSW	x	x	F_AND	x	x	MF_TGO3	x	u
MF_TGI2	x	u	F_OR	x	x			
MF_TGI2p	x	u	F_EXOR	x	x			
MF_TGI3	x	u	F_LATCH	x	x			
MF_TGI4	x	u	F_COUNT	x	x			
MF_TGI5	x	u	F_WAIT	x	x			
MF_TGI6	x	u	F_DELAY	x	x			
MF_TGIO *	x/-	u	F_TOUT	x	x			
MF_ONLINE	x	u	F_LTH	x	x			
MF_RTC	x	-						
MF_CAKS	x	-						
MF_CAKM	x	-						
MF_TGICC11_1	x	-						
MM_IA1	-	x	??					
MM_IA2	-	x						
MM_IB1	-	x						
MM_IC1	-	x						
MM_IC1a	-	x						
MM_IC1b	-	x						
MM_IC2	-	x						
MM_IC2a	-	x						
MM_IC2b	-	x						
MM_ID1	-	x						
MM_IE1	-	x						
MM_IE2	-	x						
MM_IE3	-	x						
MM_IE3x	-	x	??					
MM_IE4	-	x						
MM_IE5	-	x	??					
MM_OE1	-	x						
MM_OE2	-	x						
F_TIM	x	x						
F_DATE	x	x						

* = Input element with integrated output element
x = Element designed for an equipment type.
U = Element usable for equipment type.
\- = Element not usable in equipment type.
x/x = Diffrentiation between stand alone MF (MF7033) and internal MF (MF7013 / MF7023).

10 Element Description - HW Inputs / Outputs

All MUX and DMUX cards and lines, as well as PIA ports/lines, have predefined names. They all have a general pattern, see the specific function for furter details.

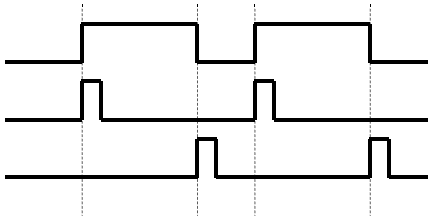
10.1 D_IN

D_IN names are build as follows:

M	MF/MM7033 (is a global signal identifier)
DI	Digital Input
x	Card Number
y	input line number
e	specifies edge of signal

The state of the switch:

Parameters :



S = follow actual state of input switch

P = reaction on positive edge (Pulse)

N = reaction on negative edge (Pulse).

Syntax:

```
instruction :
D_IN(1: GlobalInSignalIdentifier);
```

Example:

```
a_dIn :D_IN(1: M_DI_1_1_S);
```

10.2 D_OUT

D_OUT names are built as follows:
`M_DO_x_y_e or M_PIAO_x_y_e`

- M MF/MM7033 (is a global signal identifier)
- DO Digital Output DMUX
- PIAO Digital Output to PIA
- x Card Number
- y output line Number
- e describes characteristics of digital output.
  - S follow actual state of digital input.
  - B Attribute: output flashing (blinking).

'S' is always used as if it is the output itself. In additional the attribute flashing/not flashing must be allocated to the signal. This can be done in three different ways:

1) by using M_DO_x_y_B as a signal name = variable:
   - if attribute M_DO_x_y_B is active, then output M_DO_x_y_S will flash (when activated)
   - if attribute M_DO_x_y_B is inactive, then output M_DO_x_y_S will light continuously (when activated)
2) by setting attribute M_DO_x_y_B to 'K_TRUE':
   - output M_DO_x_y_S will flash whenever activated
3) by setting attribute M_DO_x_y_B to 'K_FALSE':
   - output M_DO_x_y_S will light continuously whenever activated

Syntax:

```
instruction:
D_OUT(2: GlobalOutSignalIdentifier_S,
         GlobalOutSignalIdentifier_B;
      2: Logic, Ledtest);
```

Parameters:

`GlobalOutSignalIdentifer_S:`
`M_DO_1_1_S..M_DO_16_48_S,` - actual state of the digital output
`M_PIAO_1_1_S..M_PIAO_10_8_S.` - actual state of the PIA output

`GlobalOutSignalIdentifer_B:`
`M_DO_1_1_B..M_DO_16_48_B,` - attribute flashing for DMUX
`M_PIAO_1_1_B..M_PIAO_10_8_B,` - attribute flashing for PIA output
`K_TRUE, K_FALSE.`

`Logic:`
`posLogic, negLogic.`

`LedTest:`
`noLEDtest, LEDtest.`

Example:

```
a_dOut :D_OUT(2: M_DO_1_23_S, M_DO_1_23_B; 2:posLogic, noLEDtest);
```

11 Element Description - Logic connections

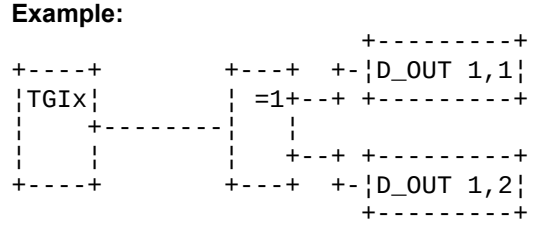
All inputs and outputs of the elements described may be connected directly to a physical output (D_OUT = DMUX), to an internal signal identifier (max. 32 characters), or be not connected (='NC'). Defining inputs as 'not connected' is a theoretical possibility, although it does not make much sense!
Inputs can be set to a fixed value:

- 'K_TRUE' means that the input is always active ('high' or '1')
- 'K_FALSE' means that the input is always inactive ('low' or '0').

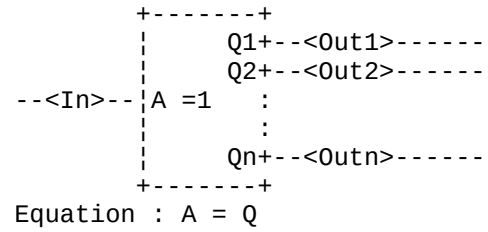
11.1 Equal

Description:
Equal has the function of a buffer. It is required when a MUX must be connected directly to a DMUX. Another feasible example of usage is when one input telegram must control several digital outputs (D_OUT).
In this case, the D_OUT name cannot be used directly!

Example:



Symbol:



Equation : A = Q

Syntax:

instruction:
F_EQUAL (1: A; y: Q{, Q});

Parameters:

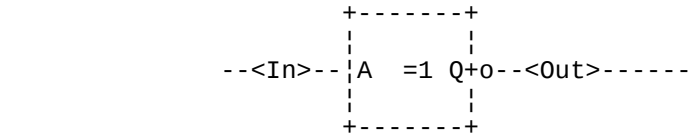
A	Link-ID of input signal	InSignal
y	number of outputs	1..1000
Q	Link-ID of output signal	OutSignal

Example:
Output is connected to a internal signal line.a_buffer :
`F_EQUAL(1: In0001; 1: Out0001);`
Output is connected to 3 D_OUTs: (DMUX card number 1 output 2, DMUX card number 2 output 2, DMUX card number 1 output 17). a_buffer :
`F_EQUAL(1: In0001; 3: M_DO_1_2_S, M_DO_2_2_S, M_DO_1_17_S);`

11.2 Not

Description:
This element is an inverter. When the input is TRUE, the output is FALSE and vice versa.

Symbol:




```
Equation : A = Q
```

Syntax:

```
instruction :
F_NOT ( 1: A; 1: Q);
```

Parameters:

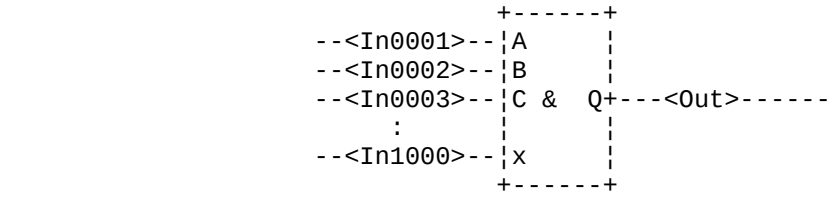
A	Link-ID of input signal	InSignal
Q	Link-ID of output signal	OutSignal

Example:

```
a_inverter :  F_NOT( 1: In0001; 1: Out0001);
```

11.3 And

Symbol:



Truth table:

Below is the truth table of a two input AND (F = A ¨ B).

A	B	F
0	0	0
0	1	0
1	0	0
1	1	1

Syntax

```
instruction :
F_AND ( y:A, B {, C}; 1:Q);
```

Parameters:

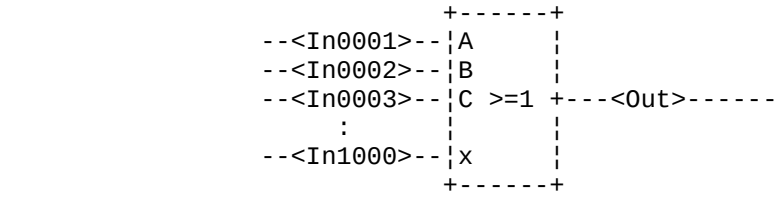
y	number of inputs	2..1000
A	Link-ID of input number A	InSignal
B	Link-ID of input number B	InSignal
C	Link-ID of input number C	InSignal
	:	
	:	
x	Link-ID of input number 1000	InSignal
Q	Link-ID of output signal	OutSignal

Example:

```
a_and_gate :F_AND ( 3: In0001, In0002, In0003; 1: Out0001);
```

11.4 Or

Symbol:



Truth table:

Below is the truth table of a two input OR (F = A + B).

A	B	F
0	0	0
0	1	1
1	0	1
1	1	1

Syntax:

```
instruction :
F_OR ( y: A, B {, C}; 1: Q);
```

Parameters:

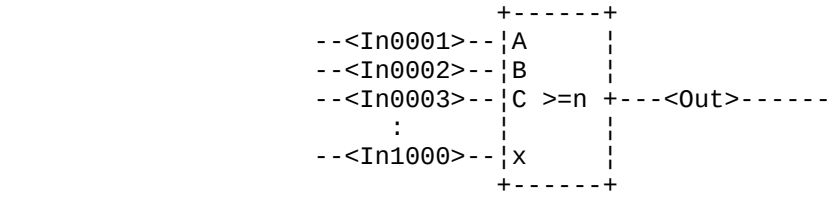
```
y  number of inputs                2..1000
A  Link-ID of input number A       InSignal
B  Link-ID of input number B       InSignal
C  Link-ID of input number C       InSignal
        :
        :
x  Link-ID of input number 1000    InSignal
Q  Link-ID of output signal OutSignal
```

Example:

```
a_or_gate :F_OR( 3: In0001, In0002, In0003; 1: Out0001);
```

11.5 LTH logic threshold

Symbol:



Truth table:

Below is the truth table of a three input LTH (logic threshold) with n = 2.

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Syntax:

```
instruction :
F_LTH ( 1: n; y: A, B {, C}; 1: Q);
```

Parameters:

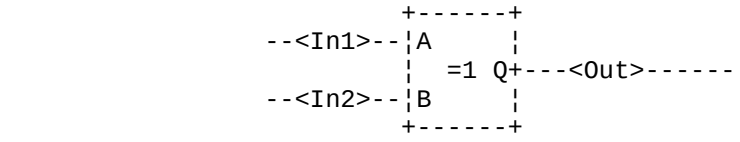
```
n  threshold                        1..y
y  number of inputs                 2..1000
A  Link-ID of input number A        InSignal
B  Link-ID of input number B        InSignal
C  Link-ID of input number C        InSignal
         :
         :
x  Link-ID of input number 1000     InSignal
Q  Link-ID of output signal OutSignal
```

Example:

```
a_2zone_dependence :    F_LTH( 1: 2;
                               9: Zone01, Zone02, Zone03,
                                  Zone04, Zone05, Zone06;
                                  Zone07, Zone08, Zone09;
                               1: Out0001);
```

11.6 EXOR

Symbol:



Truth table:

Below is the truth table of an EXOR

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

Syntax:

```
instruction :
F_EXOR ( 2: A, B; 1: Q);
```

Parameters:

```
A  Link-ID of input number A      InSignal
B  Link-ID of input number B      InSignal
Q  Link-ID of output signal Q     OutSignal
```

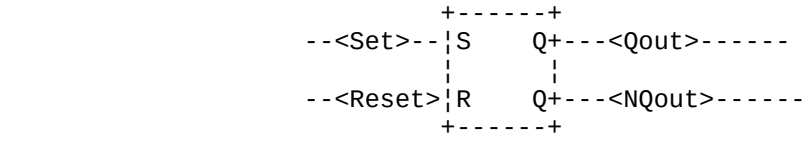
Example:

```
an_exor :F_EXOR ( 2: In0001, In0002; 1: Out0001);
```

11.7 LATCH

Description:
A latch can be used to convert a dynamic signal to a static one. It is, of course, also useful as a memory element.

Symbol:



Truth table:
Below is the truth table of a SR-latch

S	R	Q	Q
0	0	Q	Q
0	1	0	1
1	0	1	0
1	1	illegal	

Syntax:

```
instruction :
F_LATCH ( 2: Set, Reset; 2: Q, Q);
```

Parameters:

Set	Link-ID of set input	InSignal
Reset	Link-ID of reset input	InSignal
Q	Link-ID of Q output	OutSignal
Q	Link-ID of not-Q output	OutSignal

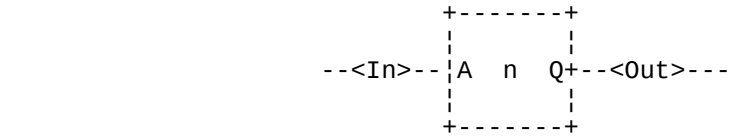
Example:

```
a_latch :F_LATCH ( 2: In0001, In0002; 1: Out0001, Out0002 );
```

11.8 Count

Description:
The output goes active (pulse) when the specified number (n) of events has occured, and the counter is reset to zero.

Symbol:



Syntax:

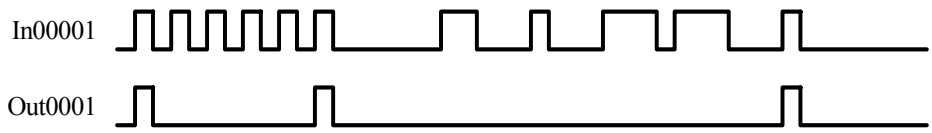
```
instruction :
F_COUNT( 1: n; 1: A; 1: Q);
```

Parameters:

n	Number of counts	1..16000000
A	Link-ID of set input	InSignal
Q	Link-ID of reset input	OutSignal

Example:

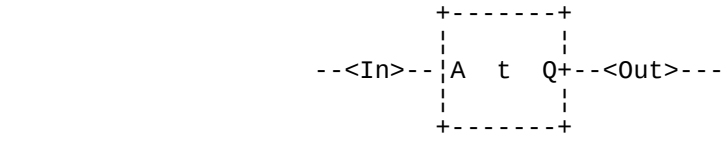
```
a_counter :F_COUNT ( 1: 5; 1: In0001; 1: Out0001);
```



11.9 Wait

Description:
The output goes active (pulse) when the time delay (t sec.) expires. The pulse length is 250ms.

Symbol:



Parameters:

t	wait time in seconds	0..16000000
A	Link-ID of input	InSignal
Q	Link-ID of output	OutSignal

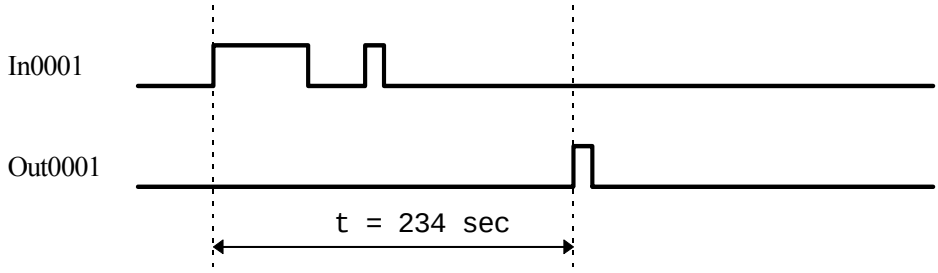
Limitations:
Max delay time : 16000000 sec

Syntax:

```
instruction :
F_WAIT ( 1: t; 1: A; 1: Q);
```

Example:

```
pause :F_WAIT ( 1: 234; 1: In0001; 1: Out0001);
```



11.10 Delay

Description:
This element has two important parameters. The first is the 'delay' parameter, this gives the waiting time until the pulse goes active. The second parameter defines the 'pulse length', which means how long the output shall be active.

Symbol:

```
                +-------------+
                |             |
        --<In>--|A del,pulse Q|--<Out>---
                |             |
                +-------------+
```

Parameters:

del	delay time in seconds	0..16000000
pulse	pulse length in seconds	0..16000000
A	Link-ID of input	InSignal
Q	Link-ID of output	OutSignal

Limitations:

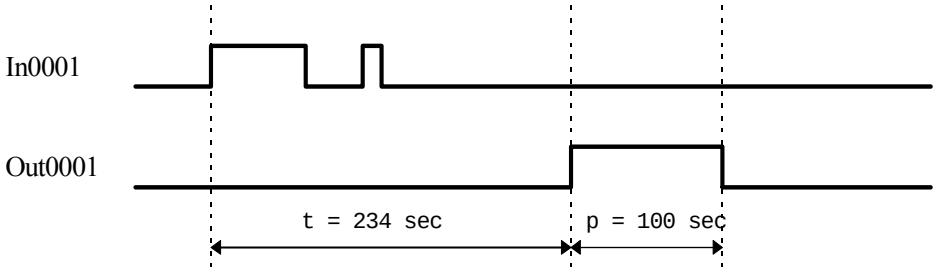
Max delay time : 16000000 seconds
Max pulse time : 16000000 seconds

Syntax:

```
instruction :
F_DELAY ( 2: del, pulse; 1: A; 1: Q);
```

Example:

```
pause :F_DELAY ( 2: 234, 100; 1: In0001; 1: Out0001);
```



11.11 Tout

Description:
The timer is started when 'Set' goes high. 'Clr' resets the timer. If 'stop' goes high within time 't', then output 'Stopped' will go high. If nothing happens within the time 't' after 'Set' went high, the output 'Expired' will go high. The timer always start from 0, when 'Set' goes high, even if 'Clr' or 'Stop' was activated before timeout.

Symbol:

```
                +--------------+
        --<set>----¦Set    Expired+--<expired>---
        --<clear>--¦Clr t=x sec   ¦
        --<stop>---¦Stop   Stopped+--<stopped>---
                +--------------+
```

Parameters:

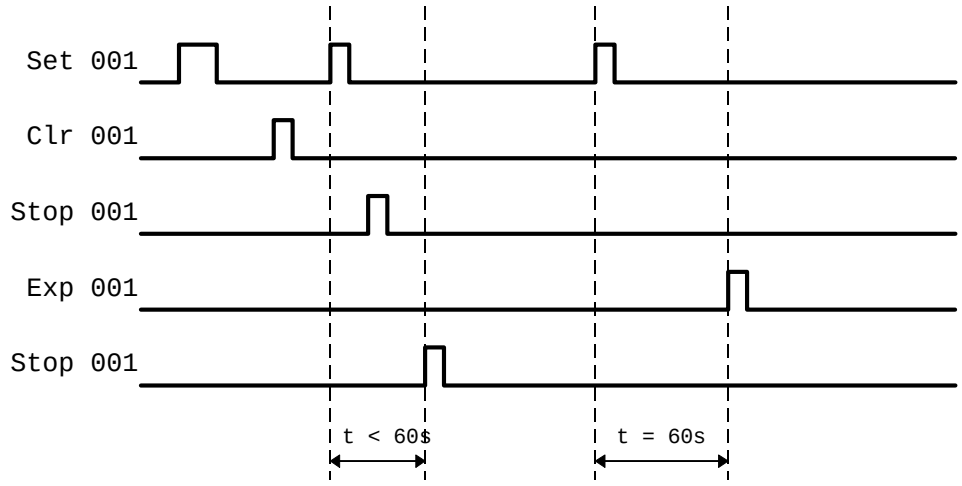
t	time for timeout (in seconds)	2..16000000
set	Link-ID of set input	InSignal
Clr	Link-ID of reset input	InSignal
Stop	Link_id of stop input	InSignal
Expired	Link-ID of expired output	OutSignal
Stopped	Link-ID of stopped output	OutSignal

Syntax:

```
instruction :
F_TOUT( 1: t; 3: set, clr, stop; 2: expired, stopped);
```

Example:

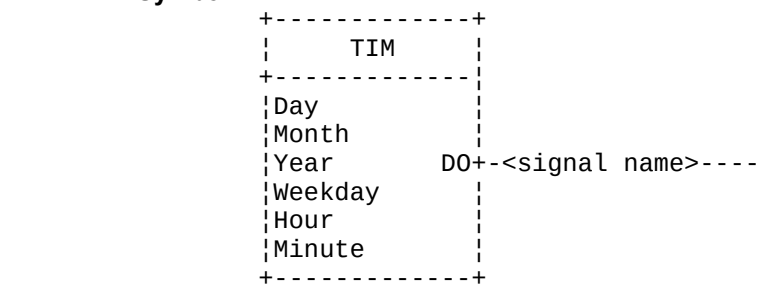
```
timeout : F_TOUT ( 1: 60; 3: Set001, Clr001, Stop001;
                   2: Exp001, Stop001);
```



11.12 Tim

Description:
'Tim' is a timer element. It generates a pulse at the programmed time. The pulse length is 1 minute.

Symbol:



Syntax:

```
instruction :
F_TIM( 6: Day, Month, Year, Weekday, Hour, Minute;1: DO);
```

Parameters:

Day		1 .. 31, *
Month		1 .. 12, *
Year		0 .. 99, *
Weekday		mon, tue, wed, thu, fri, sat, sun, *
Hour		0 .. 23, *
Minute		0 .. 59,
DO	ink-ID of output	OutSignal

'*' is a wildcard (e.g. Day = * equals any day)

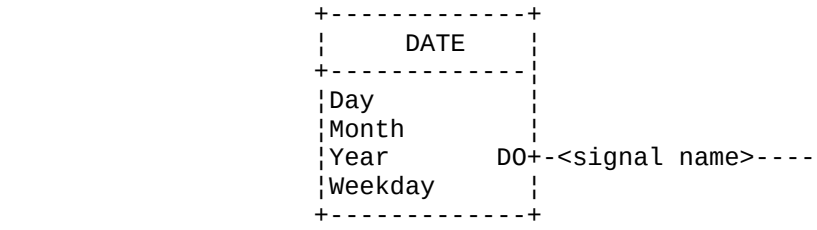
Example:

```
a_timer :F_TIM (6: 14, 7, *, fri, 17, 0; MainSwitchOffSummerVa-
cation);
```

11.13 Date

Description:
'Date' is a timer element. It generates a pulse of 24 hours at the programmed date.

Symbol:



Syntax:

```
instruction :
F_DATE( 4: Day, Month, Year, Weekday;1: DO);
```

Parameters:

Day		1 .. 31, *
Month		1 .. 12, *
Year		0 .. 99, *
Weekday		mon, tue, wed, thu, fri, sat, sun, *
DO	Link-ID of output	OutSignal

'*' is a wildcard (e.g. Day = * equals any day)

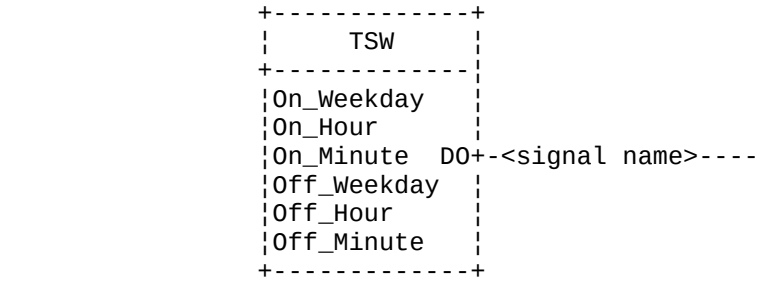
Example:

```
a_holiday :F_DATE (6: 14, 7, *, *; MainSwitchOffSummerVacation);
```

11.14 TSW

Description:
'Tsw' is a time switch. It is active from the programmed start time to the programmed stop time (static output).

Symbol:



Syntax:

```
instruction :
F_TSW( 6: On_weekday, On_Hour, On_minute,
          Off_weekday, Off_Hour, Off_minute;
       1: DO);
```

Parameters:

On_Weekday	mon, tue, wed, thu, fri, sat, sun, *
On_Hour	0 .. 23
On_Minute	0 .. 59
Off_Weekday	mon, tue, wed, thu, fri, sat, sun, *
Off_Hour	0 .. 23
Off_Minute	0 .. 59
DO Link-ID of output	OutSignal

'*' is a wildcard (e.g. Weekday = * equal any day of the week)

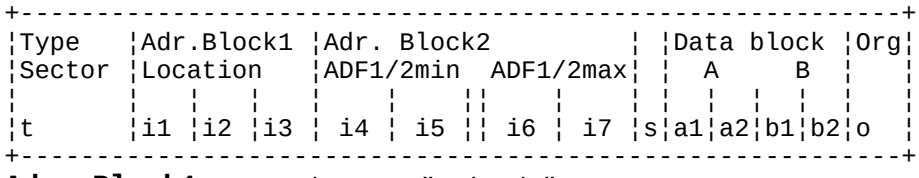
Example:

```
a_time_switch :F_TSW (6: mon, 6, 15, fri, 16, 0;
 1: CoffeeMachineOnOff);
```

12 Element description - MF7033

12.1 Construction of telegrams for report and control

The globally valid element parameters are defined below. Most of the MF elements are constructed according to the shown figure. Some elements need special parameters, they are described in the decription of the actual element. Details of valid telegrams are described in the handbook AH0.1.



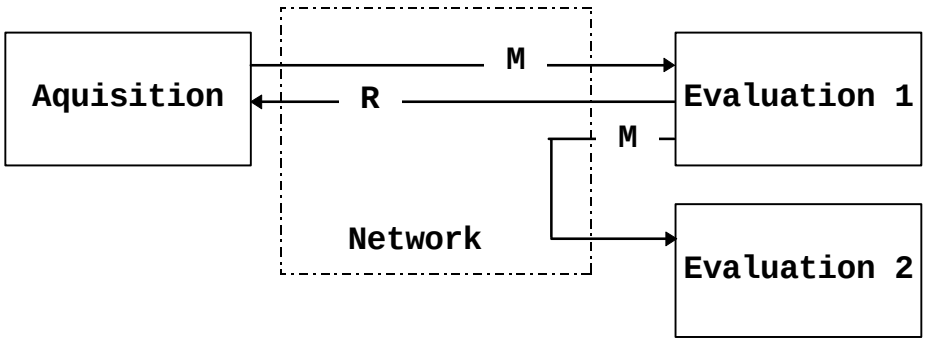
Adr. Block1: Location "UnitAdr".
i1..i3 100,110..118
120..128, .., 240..248
800..862, 900

Type: Sector "Sect"
. t A,B,C,D,E,F,0

Adr.Block2: i4-i5 ADF1Min
i6-i7 ADF2Min
. > 00..FD
i4-i5 ADF1Max
i6-i7 ADF2Max

Data Block: A "DataA" 00..FD
B "DataB" 00..FF (wildcard)

s - Direction block:
The letter 's' in the figure above is the direction block. The basic principles of this block is shown in the figure below.



- **M** = message
  - **U** = unacknowledged
  - **Q** = acknowledged
  - **N** = unacknowledgeable
- **R** = command

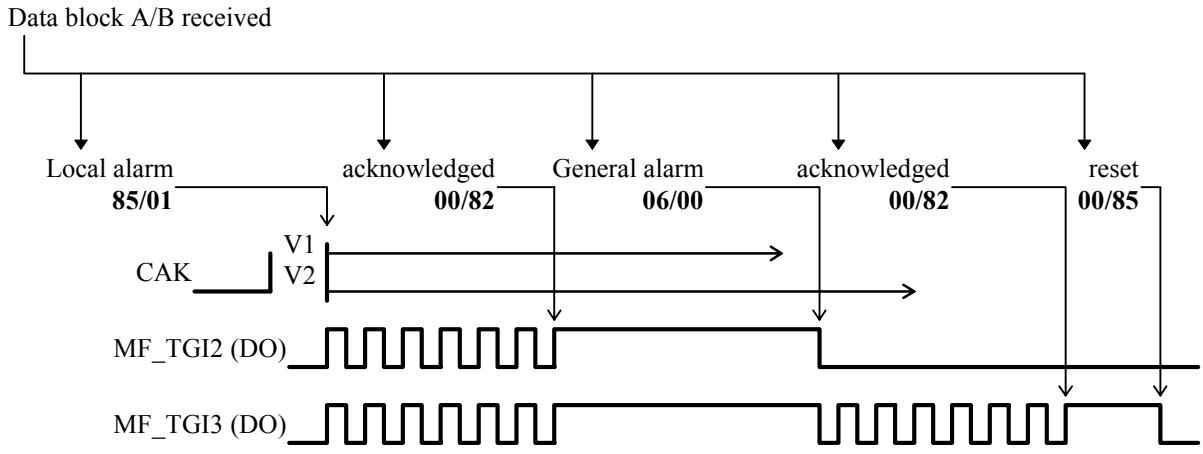
o - Organisation:
day/night: Evaluation of the telegrams is dependent of the current alarm organisation.

anyOrg : Evaluation of the telegrams is not dependent of the current alarm organisation.

12.2 Telegram Input Elements

To activate/deactivate an output of a element, the incoming telegram must match the data defined in the parameter list, or be within the defined data range for the element. All outputs of the input elements may have the parameter not connected = 'NC'. The outputs of a TGIx can be connected directly to a
D_OUT (DMUX line). D_OUT have predefined signal names. Outputs of internal signals may also be given names, which can be used for further connections to logic elements or to TGO-elements.
Generally a CZ10 with V1/V2 sends two alarms when it is in organization day. These are local alarm and general alarm.
The local alarm is transmitted with the zone information, but the general alarm sends no zone information (a so called collective telegram). Since all input elements for an MF have section/zone information, the correct reaction upon collective telegrams is not in any case secured. But for the transition from local alarm to general alarm the MF works the following way:
All allready active zones in a control unit (CZ10 sector fire) are internal updated when a general alarm is received. If the zones was acknowledged local alarms, the counters for the corresponding zones go to unacknowledged general alarms. Unacknowledged local alarms are transited to unacknowledged general alarms.
Please observe that the above explained matter is reflecting the internal life of the process image. The visual reaction is element dependent. Only elements that evaluates any type of alarm may be used, e.g MF_TGI3 with the parameter 'anyAlarm', or MF_TGI4 with the status parameter 'alarm'.
The outputs of the elements are described below. The output DO' is shown with the attribute flashing active until the message is acknowledged.

```
:MF_TGI2 (9:111,A,A1,01,A1,05,85,01,day;
      3:M_DO_1_1_S,M_DO_1_1_B,NC);
:MF_TGI3 (9:111,A,A1,01,A1,05,anyAlarm,automatic,day;
      3:M_DO_1_1_S,M_DO_1_1_B,NC);
```



Process image dependent MF elements do not distinguish between a first alarm and a subsequent alarm.

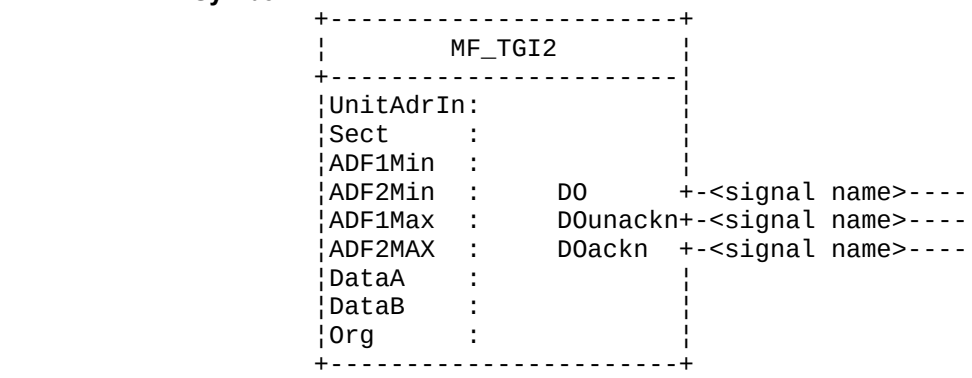
12.3 Data block A, B stored in the process image

AH0.1 is the reference table for telegrams. Most of the defined telegrams in AH0.1 have a number connected to it, this indicates to witch counter in the DMS-system that the telegram is internally connected. Only the telegrams that have this number defined is stored in the process image of the MF. Thus it is not possible to use counter oriented MF elements for data A/B combinations other than those defined with a counter number in AH0.1.

12.3.1 MF_TGI2

Description:
The MF_TGI2 is the standard telegram input element. It is counter oriented following the process image. The parameters ADF1 and ADF2 can be defined in ranges. The output is static, which means that 'DOunackn' is active as long as one (or more) of the technical addresses defined in the ADF1/ADF2 range are active, but not acknowledged. The output signal'DOackn' is active as long as one (or more) of the technical addresses defined in ADF1/ADF2 range are active and acknowledged.
Note that Both 'DOunackn' and 'DOackn' can be active at the same time if a range of data is defined. If ADF1Min = ADF1Max and ADF2Min = ADF2Max, the output DO and only one of the other two outputs Dounackn and DOackn will be active at the same time. The output DO is an internal 'or' connection of the two outputs DOunackn and DOackn.
If an incoming telegram matches the address (ADF1/ADF2) and the data range (data A/B) it is stored in the process image and the signal DO is set to high (TRUE). This situation lasts until the match is no longer fulfilled whereupon normal operation will be resumed.
The parameter DataB = FF is a wildcard. If FF is set, TGI2 reacts on any data B received.

Symbol:



Syntax:

```
instruction :
MF_TGI2 ( 9: UnitAdrIn, Sect, ADF1Min, ADF2Min,
             ADF1Max, ADF2Max, DataA, DataB, Org;
          3: DO, DOunackn, DOackn);
```

Parameters:

UnitAdrIn	Location	100, 110..118, 120..128,...,240..248, 800..862, 900.
Sect	Sector	A, B, C, D, E, F, 0.
ADF1Min	Address field 1 min.	00..FD.
ADF2Min	Address field 2 min.	00..FD.
ADF1Max	Address field 1 max.	00..FD.
ADF2Max	Address field 2 max.	00..FD.
DataA	Data block A	00..FD.
DataB	Data block B	00..FF. (FF = wildcard)
Org	Organisation	day, night, anyOrg.
DO	Link-ID of DO	OutSignal.
DOunackn	Link-ID of DOunackn	OutSignal.
DOackn	Link-ID of DOackn	OutSignal.

Example:

```
a_tgi2_1 :MF_TGI2 ( 9: 111, A, A1, 01, A1, 24, 85, 01, anyOrg;
                    3: OutDO001, OutUnAckn001, OutAckn001);
a_tgi2_2 :MF_TGI2 ( 9: 112, A, 01, 01, 01, 50, 67, 4F, anyOrg;
                    3: OutDO002, OutUnAckn002, OutAckn002);
a_tgi2_3 :MF_TGI2 ( 9: 113, A, A1, 05, A1, 05, 85, 01, anyOrg;
                    3: NC, M_DO_1_1_S, M_DO_4_23_S);
```

In 'a_tgi2_1' and 'a_tgi2_2' the outputs have internal signal names, they can be used to connect to other elements (logic or TGOx). The third element 'a_tgi2_3' has one output open (NC). Outputs 'DOunacn' and 'DOackn' are connected directly to DMUX card number 1 line 1 and DEMUX card number 4 line 23.

12.3.2 MF_TGI2p

Description:

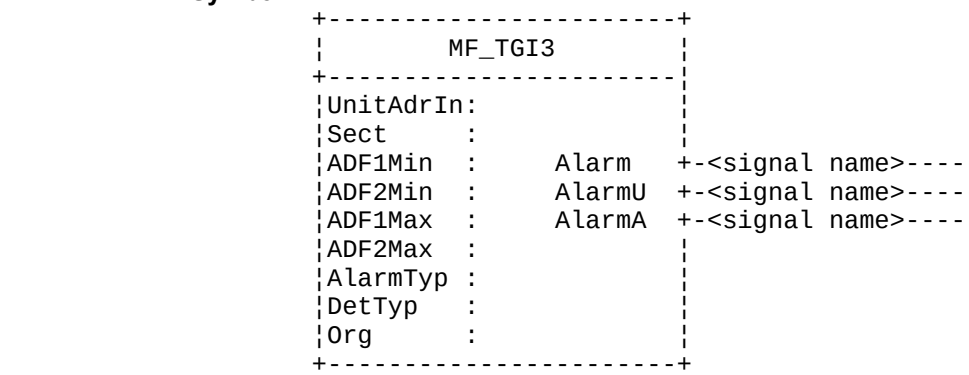
MF_TGI2p is equivalent to MF_TGI2, except the output characteristic.
The outputs are not static like it is for MF_TGI2. On event the output DO and DOunackn will go high for 250ms, and then low again. Thus the 'p' in MF_TGI2p means 'pulse'. After acknowledge of the event, a pulse is set on the outputs DO and DOackn.
The pulse length may get stretched (> 250ms), if more events to the same address range occur at allmost the same time.

➜ See MF_TGI2 for syntax, parameters and examples.

12.3.3 MF_TGI3

Description:
The MF_TGI3 is an element specially dedicated to the easy handling of alarm telegrams. The element is counter oriented. Two special parameters can be defined - AlarmType and DetectorType.
Alarm type functions like a filter for different types of alarm (local alarm/general alarm etc.). The table below decribes the connection between the sector and the corresponding alarm type.
Detector type has a similar filter function, but filters out the different types of detectors (automatic / push button etc.). See the table below for connection detector type - alarm - valid sector. If an incoming telegram matches the specified address range, the datablock A alarm range (01..0F, 81..8F) and the alarmType - detector- type (specified or wildcarded), then the "Alarm"-Signal is activated.
Depending on the acknowledgement state of the received telegram, either the "AlarmU"- or the "AlarmA"-Signal is also activated. These outputs are static, which means that they remain active as long as the received telegrams satisfy the specified parameters.

Symbol:



Syntax:

```
instruction :
MF_TGI3 ( 9: UnitAdrIn, Sect, ADF1Min, ADF2Min,
             ADF1Max, ADF2Max, AlarmTyp, DetTyp, Org;
          3: Alarm, AlarmU, AlarmA );
```

AlarmType	Data block A	Standard text	
alarm1	85/05	Local alarm	A, E
alarm2	86/06	General alarm	A, E
anyAlarm	85/05/86/06	----------	A, E
alarm1	84/04	Pre alarm	B
alarm2	88/08	Extinction alarm	B
anyAlarm	84/04/88/08	----------	B
alarm1	81/01	Alarm	C
alarm2	8C/0C	Sabotage	C
anyAlarm	81/01/8C/0C/8D/0D	Alarm/Sabotage/Alarm+Sabo	C
alarm1	82/02	Prewarning	D
alarm2	87/07	Prewarning	D
anyAlarm	82/02/87/07	----------	D

DetType	Data block A	Data block B = DetType	Valid Sectors
general	01..0F, 81..8F	00	A, B, D, E
automatic	01..0F, 81..8F	01	A, E
manual	01..0F, 81..8F	02	A, E
burglary	01..0F, 81..8F	05	C
holdup	01..0F, 81..8F	06	C
theft	01..0F, 81..8F	07	C
keyingError	01..0F, 81..8F	08	C
timeLock	01..0F, 81..8F	09	C
controlUnit	01..0F, 81..8F	0A	C
threat	01..0F, 81..8F	0C	C
anyDet	01..0F, 81..8F	(detector type wildcard)	A, B, C, D, E

Parameters:

UnitAdrIn	Location	100, 110..118, 120..128,...., 240..248, 800..862, 900.
Sect	Sector	A, B, C, D, E, F.
ADF1Min	Address field 1 min.	00..FD.
ADF2Min	Address field 2 min.	00..FD.
ADF1Max	Address field 1 max.	00..FD.
ADF2Max	Address field 2 max.	00..FD.
AlarmType	Type of alarm	see table above.
DetType	Type of detector	see table above.
Org	Organisation	day, night, anyOrg.
Alarm	Link-ID of Alarm	OutSignal.
AlarmU	Link-ID of AlarmU	OutSignal.
AlarmA	Link-ID of AlarmA	OutSignal.

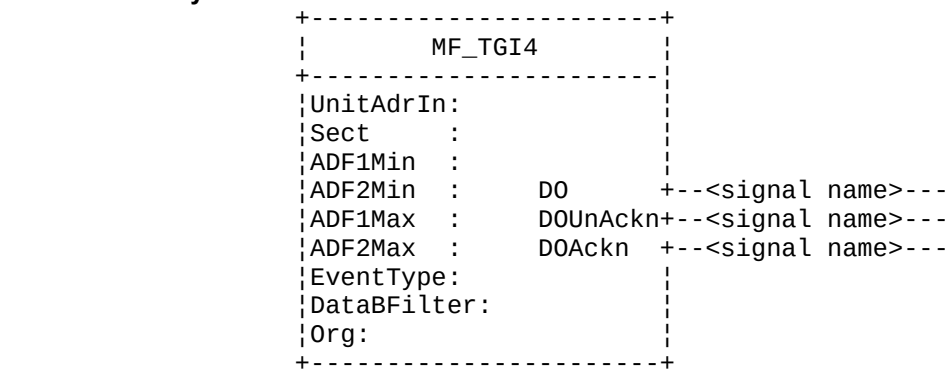
Example:

```
a_tgi3_1 :MF_TGI3 ( 9: 111, A, A1, 01, A1, 09, alarm1, manual, anyOrg;
                    3: OutAlarm001, OutAlarmU001, OutAlarmQ );
```

12.3.4 MF_TGI4

Description:
TGI4 is an element specially dedicated to the easy handling of status telegrams. It is counter oriented. As an additional feature it is possible to include an additional filter for dataB. DataBFilter = 'FF'means NO FILTER.
If an incoming telegram is in the defined address range and dataA, dataB corresponds to the status message defined in the PLC file - then the one of the outputs will be activated, depending on whether the received telegram was acknowledged or not. The output DO will go active, as there is a logical OR connection between the two other outputs.
All outputs are static, which means that they remain active until the internal status counter for unacknowledged (or acknowledged) messages goes to zero.

Symbol:



Syntax:

```
instruction :
MF_TGI4 ( 9: UnitAdrIn, Sect, ADF1Min, ADF2Min,
             ADF1Max, ADF2Max, EventType, DataBFilter, Org;
          3: DO, DOUnAckn, DOAckn);
```

Parameters:

UnitAdrIn	Location	100, 110..118, 120..128,..., 240..248, 800..862, 900.
Sect	Sector	A, B, C, D, E, F, 0.
ADF1Min	Address field 1 min.	00..FD.
ADF2Min	Address field 2 min.	00..FD.
ADF1Max	Address field 1 max.	00..FD.
ADF2Max	Address field 2 max.	00..FD.
EventType	Type of event	others, off, test, active, warning, fault, alarm, drift.
DataBFilter	Filter of data B ('FF'=wildCard - any data B)	00..FF
Org	Organisation	day, night, anyOrg.
DO	Link-ID of DO	OutSignal.
Dounackn	Link-ID of Dounackn	OutSignal.
Doackn	Link-ID of Doackn	OutSignal.

Example:

```
atgi4_1 :MF_TGI4 ( 9: 213, 0, 00, 00, 00, 0F, fault, FF, anyOrg;
                   3: M_DO_2_31_S, M_DO_2_33_S, M_DO_2_36_S);
```

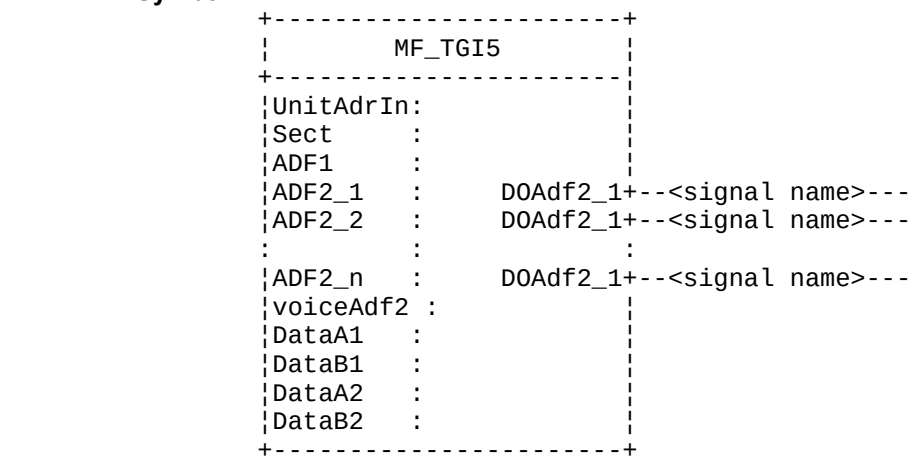
The example above evaluates ALL troubles in the basic sector, location 213.
Note that basic sector is NOT organisation dependent, thus evaluation of troubles for organisation day only is meaningless.

12.3.5 MF_TGI5

Description:
TGI5 is specially dedicated to VOICE applications, but it might also be handy in some other applications. This element is not process image oriented. TGI5 reacts on dataA and dataB directly. Its output is activated by the occurence of data A1,B1. When data A2,B2 occurs, the output is deactivated.
A special type of wildcard is allowed for dataA2, B2, this is dataA2=FF, and dataB2=FF. When a telegram is received where dataA and dataB differ from the defined parameters A1 and B1, the output switches to inactive. Of course the sector, location, ADF1 and ADF2 in the received telegram must match parameters defined in the actual TGI5.

Voice Applications:
TGI5 contains a list of ADF2. Each of these ADF2 has separate outputs. When an incoming telegram has the same ADF2 as one of those defined in the parameter list, the corresponding output will switch according to the data A and B in the received telegram. If a 'compact telegram' of the type page=BC, evac=BD or alert=BE occurs, all outputs defined in the parameter list will switch according to data A and B of the received telegram.

Symbol:



Syntax:

```
instruction :
MF_TGI5 ( 3: UnitAdrIn, Sect, ADF1;
          n: ADF2_1, ADF2_2, , ADF2_n;
          5: voiceAdf2, DataA1, DataB1, DataA2, DataB2;
          n: DO_1, DO_2, ...., DO_n);
```

Parameters:

```
UnitAdrIn          Location           100, 110..118,
.                                     120..128,...., 240..248,
.                                     800..862, 900.
Sect               Sector             A, B, C, D, E, F, 0.
ADF1               Address field 1    00..FD.
ADF2_1..ADF2_n     Address field 2    00..FD.
VoiceAdf2          Address field 2 = voice page          ADF2=BC)
.                  Address field 2 = voice evac          ADF2=BD)
.                  Address field 2 = voice alert         ADF2=BE)
.                  Address field 2 = voice noVoice (ADF2=not BC,BD or BE)
DataA1       Data block A 00..FD.
DataB1       Data block B 00..FD.
DataA2       Data block A  - voice    00..FF. ('FF' = wildCard)
DataB2       Data block B  - voice    00..FF. ( 'FF' = wildCard )
DO_1..DO_n          Link-ID of DO_1 to DO_n OutSignal.
```

Example:

```
atgi5_1 :MF_TGI5 ( 3: 213, E, E1; 5: 49, 51, 53, 55, 61;
                   5: page, 41, 4F, 41, 3C;
                   5: M_DO_1_1_S, M_DO_1_2_S, M_DO_1_3_S, M_DO_1_4_S,
                      M_DO_1_5_S, );

atgi5_2 :MF_TGI5 ( 3: 213, E, E1; 3: 50, 52, 54;
                   5: evac, 41, 4F, 41, 3C;
                   3: M_DO_2_1_S, M_DO_2_2_S, M_DO_2_3_S);

atgi5_3 :MF_TGI5 ( 3: 215, E, E1; 3: 69, 70, 71;
                   5: alert, 41, 4F, 41, 3C;
                   3: M_DO_3_1_S, M_DO_3_2_S, M_DO_3_3_S);

atgi5_4 :MF_TGI5 ( 3: 111, C, 00; 1: EF; 5: noVoice, 55, 60, 55, 61;
                   1: M_DO_1_1_S);
```

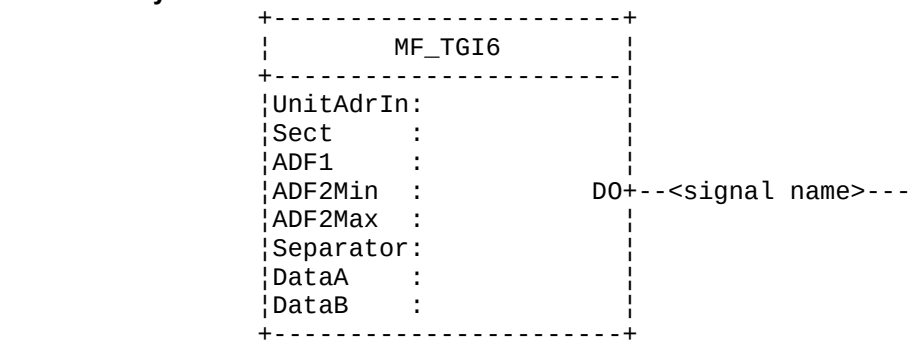
The number of ADF2 entries and the number of DOAdf2 MUST ALWAYS be the same. See example above (...x:..).

12.3.6 MF_TGI6

Description:
TGI6 is not process image oriented. The element reacts upon any valid telegram received via the communication interface, being within the specified range of address field 1 and 2. The output generated by the element is a pulse (~250ms).

The separator 'R' is only valid for the MF's own location. It is not possible to react upon a telegram that is a command to a control unit with location number 111, when the location number of the MF is for example 801. The reason for this is obvious – a telegram addressed to the unit 111 with a command to do something, is not of any interest to the other members of the system. The location 000 makes sense only with separator 'R'.

Symbol:



Syntax:

```
instruction :
MF_TGI6 (  8: UnitAdrIn, Sect, ADF1, ADF2Min, ADF2Max,
              Separator, DataA, DataB;
           1: DO
```

Parameters:

UnitAdrIn	Location	000, 100, 110..118,
.		120..128,...., 240..248,
.		800..862, 900.
Sect	Sector	A, B, C, D, E, F.
ADF1	Address field 1 min.	00..FD.
ADF2Min	Address field 2 min.	00..FD.
ADF2Max	Address field 2 max.	00..FD.
Sep	Separator	R, M, N, U, Q.
DataA	Data block A	00..FF. ('FF' = wildCard)
DataB	Data block B	00..FF. ('FF' = wildCard)
DO	Link-ID of the output	OutSignal.

Example:

```
atgi6 :MF_TGI6 ( 8: 213, E, E1, 5, 49, M, 41, 4F;
                 1: M_DO_1_1_S);
```

This configuration gives a pulse on the output on card 1,line 1 of the DMUX, when a valid telegram is received.

12.3.7 MF_TGICC11_1

Description:

The MF_TGICC11_1 is an element specially dedicated to the easy handling of CC11 alarmtelegrams. The element reacts on any valid telegram received via the communication interface. Two special parameters can be defined. "Section1..4" and "IntHorn1..4".

The CC11 has 4 Centralsection to choose:
A1/A2/A3/A4 (Central 1)
A5/A6/A7/A8 (Central 2)
A9/AA/AB/AC (Central 3)
AD/AE/AF/A0 (Central 4)

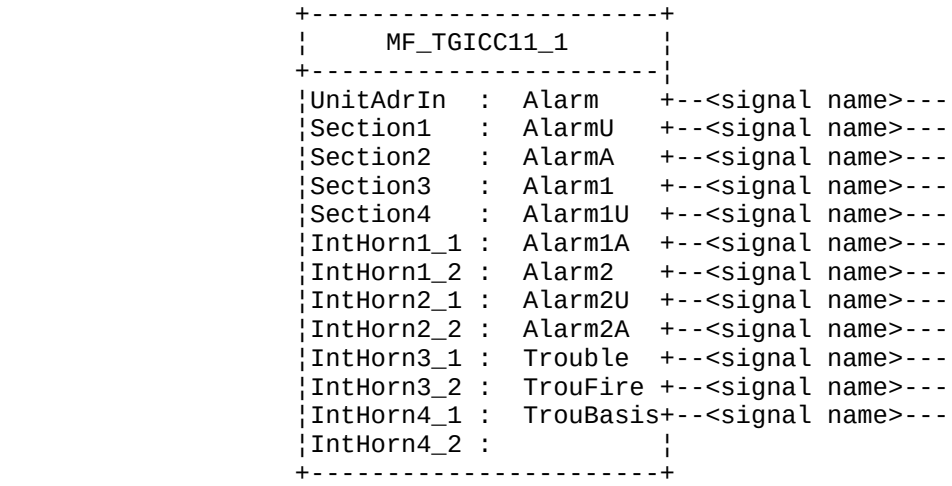
If only one "Section" is selected, the other "Section1..4" must have the same number.
Example: only "Section" A5 active = "Section1..4" has the value A5

If only one "IntHorn" is selected, the other "IntHorn1..4" must have the same number.
Example: only "IntHorn" 2 active = "IntHorn1..4" has the value "0286"

The "IntHorn1..4" inputs are used to control the "acknowleged Alarms" and "unacknowleged Alarms" outputs.
IntHorn active (7A4F) = unacknowleged Alarm
IntHorn inactive (7A4D) = acknowleged Alarm

The outputs are static, which means that they remain active as long as the received telegrams satisfy the specified parameters.

Symbol:



Syntax:

```
instruction :
MF_TGICC11_1 (13: UnitAdrIn,
                  Section1, Section2,
                  Section3, Section4,
                  IntHorn1_1, IntHorn1_2,
                  IntHorn2_1,  ¦
                  IntHorn2_2, IntHorn3_1,
                  IntHorn3_2,  ¦
                  IntHorn4_1, IntHorn4_2;
              12: Alarm, AlarmU, AlarmA,
                  Alarm1, Alarm1U, Alarm1A,
                  Alarm2, Alarm2U, Alarm2A,
                  Trouble, TrouFire, TrouBasis);
```

Parameters:

UnitAdrIn	Location	111..118, 121..128, 131..138, 141..148.
Section1..4	Centralsection	A0..AF.
IntHorn1_1..4_2	Intern Horn telegram	00..FF.
Alarm	Link-ID of Alarm	OutSignal.
AlarmU	Link-ID of AlarmU	OutSignal.
AlarmA	Link-ID of AlarmA	OutSignal.
Alarm1	Link-ID of Alarm1	OutSignal.
Alarm1U	Link-ID of Alarm1U	OutSignal.
Alarm1A	Link-ID of Alarm1A	OutSignal.
Alarm2	Link-ID of Alarm2	OutSignal.
Alarm2U	Link-ID of Alarm2U	OutSignal.
Alarm2A	Link-ID of Alarm2A	OutSignal.
Trouble	Link-ID of Trouble	OutSignal.
TrouFire	Link-ID of TrouFire	OutSignal.
TrouBasis	Link-ID of Troubasis	OutSignal.

Example:

```
a_tgicc11_1_1 :MF_TGICC11_1 (13: 111,
                                 A1, A2, A3, A4,
                                 02, 79, 02, 86, 02, 92, 02, 99;
                             12: M_DO_1_1_S, M_DO_1_2_S, M_DO_1_3_S,
                                 M_DO_1_4_S, M_DO_1_5_S, M_DO_1_6_S,
                                 M_DO_1_7_S, M_DO_1_8_S, M_DO_1_9_S,
                                 M_DO_1_10_S, M_DO_1_11_S, M_DO_12_S);

a_tgicc11_1_2 :MF_TGICC11_1 (13: 111,
                                 A1, A2, A3, A4,
                                 02, 79, 02, 86, 02, 92, 02, 99;
                             12: M_DO_1_1_S, NC, NC,
                                 NC, NC, NC,
                                 M_DO_1_7_S, M_DO_1_8_S, M_DO_1_9_S,
                                 NC, NC, M_DO_12_S);
Not all Outputs connected.

a_tgicc11_1_3 :MF_TGICC11_1 (13: 111,
                                 A2, A2, A2, A2,
                                 02, 79, 02, 86, 02, 92, 02, 99;
                             12: M_DO_1_1_S, M_DO_1_2_S, M_DO_1_3_S,
                                 M_DO_1_4_S, M_DO_1_5_S, M_DO_1_6_S,
                                 M_DO_1_7_S, M_DO_1_8_S, M_DO_1_9_S,
                                 M_DO_1_10_S, M_DO_1_11_S, M_DO_12_S);
Only centralsection "A2" active.
All "IntHorn1..4" inputs active.

a_tgicc11_1_4 :MF_TGICC11_1 (13: 111,
                                 A2, A2, A2, A2,
                                 02, 86, 02, 86, 02, 86, 02, 86;
                             12: M_DO_1_1_S, M_DO_1_2_S, M_DO_1_3_S,
                                 M_DO_1_4_S, M_DO_1_5_S, M_DO_1_6_S,
                                 M_DO_1_7_S, M_DO_1_8_S, M_DO_1_9_S,
                                 M_DO_1_10_S, M_DO_1_11_S, M_DO_12_S);
Only centralsection "A2" active.
Only "IntHorn2" input active.
```

12.3.8 MF_TGIO

Description:

TGIO is not process image oriented. On detection of a specified input telegram, a specified output telegram will be generated (together with an optional ~250ms impulse). The element reacts on any valid telegram received via the communication interface. All parameters may be wildcarded (*). Additionally it is also alowed to define the unit number, address field 1 and as ranges. The range notation is two points e.g. 12..34. The output generated by the element is a telegram plus an optional impulse (~250ms). If the impulse is not used, it must simply be defined as not connected (NC).

The wildcard mechanism works as follows:

6. Input telegram recognition

- agreement is fulfilled when all parameters in the input telegram definition agree with the corresponding data in the incoming telegram
- only parameters which are **not** specified as wildcards will be compared (ie wildcarded parameters always agree)

7. Output telegram generation

- all wildcarded parameters in the output telegram definition are replaced with the corresponding parameter values in the received telegram before transmission

The digital output is activated at the same time as the telegram is transmitted and returns to the inactive state in the next cycle (approx. 250ms after activation).

This element may not be used in an internal MF7000 (MF7013 / MF7023).
The MF does not distinguish between a first alarm and a following alarm.
The data block for a first alarm is mapped internally to the data block for the subsequent alarm. Thus, for example, you will get no response to a definition of 85 01 (local alarm, automatic detector - sector fire).
In this case you must define the data block A/B to be 05/01. The location 000 makes sense only with separator 'R'.

Limitations:

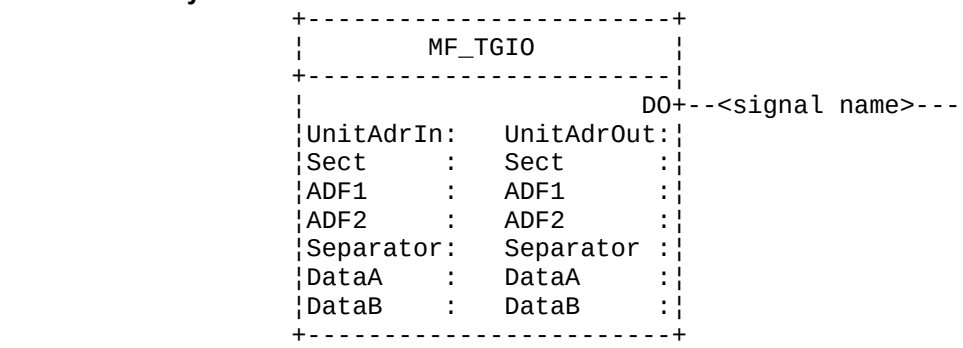
- 'R' telegrams addressed to other units are not evaluated by the MF.

The following telegrams cannot be evaluated:

- Polling on (53 55) of the own location
- Time telegrams

A output telegram configured with data block A = 00 and data block B = 00 will not be transmitted. Though it is a syntactical allowed configuration.
The usage is to activate only the output signal (DO) without sending a telegram.

Symbol:



Syntax:

```
instruction :
MF_TGIO (  7: UnitAdr[UnitAdr], Sect,
              ADF1[..ADF1], ADF2[..ADF2], Separator,
              DataA, DataB;
           7: UnitAdr, Sect, ADF1, ADF2, Separator,
              DataA, DataB;
           1: DO);
```

Parameters:

UnitAdr	Location	*, 000, 100, 110..118, 120..128,..., 240..248, 800..862, 900.
Sect	Sector	*, A, B, C, D, E, F, 0.
ADF1	Address field 1 mi	*, 00..FD.
ADF2	Address field 2 min	*, 00..FD.
Sep	Separator	*, R, M, N, U, Q.
DataA	Data block A	*, 00..FD.
DataB	Data block B	*, 00..FD.
DO	Link-ID of the output	OutSignal, NC.

Example:

```
atgio1 :MF_TGIO ( 7: *, C, 00, EF, M, 55, 60;
                  7: *, E, E1, 02, M, 41, 3C;
                  1: M_DO_1_1_S);
```

This configuration causes a pulse on the DMUX output (card 1, line 1), when a valid telegram is received. All parts of the output telegram which are wildcarded are replaced by the corresponding data from the received telegram.

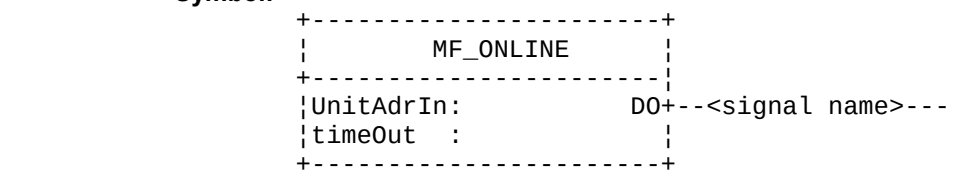
```
atgio2 :MF_TGIO ( 7: 111..114, D, D1..D8, EF, M, 3A, 3B;
                  7: 110, 0, 00, 00, M, 00, 00;
                  1: M_DO_1_1_S);
```

The atgio2 configuration evaluates all locations 111 to 114 in the section D1 to D8 for trouble end signals (3A 3B). The output (M_DO_1_1_S) is activated (pulse) when a telegram within the specified range is received. It is not sent out a telegram because the data block A and B of the output telegram is set to zero (00 00).

12.3.9 MF_ONLINE

Description:
MF_ONLINE is a element for evaluation of the communication state of the other units connected in the CERLOOP or CERBAN. The output is active as long as the communication to the defined unit is OK. The output is static and it remains active until communication is lost.

Symbol:



Syntax:

```
instruction :
MF_ONLINE ( 2: UnitAdrIn, timeOut);
            1: DO);
```

Parameters:

UnitAdrIn	Location	100, 110..118,
.		120..128,...., 240..248,
.		800..862, 900.
TimeOut		40..90 seconds.
DO	Link-ID of DO	OutSignal.

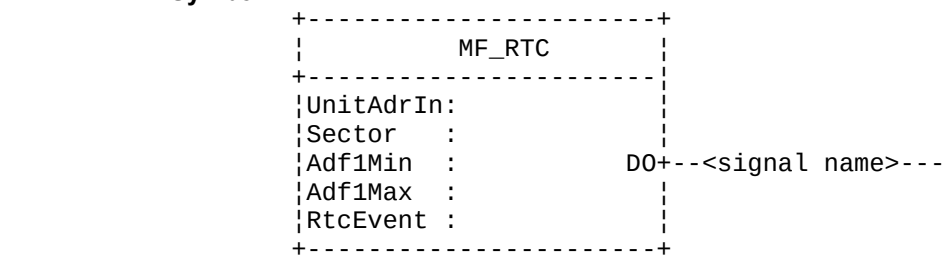
Example:

```
a_mf_online :MF_ONLINE ( 2: 111, 50;
                         1: M_DO_1_1_S);
```

12.3.10 MF_RTC (Remote Transmission)

Description:
The MF_RTC element is monitoring the remote transmission flag for troubles and alarms. The MF7000 has two flags for each control unit defined - one for troubles and one for alarms. The flag for alarm remote transmission is set when the counter for alarms is higher than 0 (1 or more alarms is already existing) and the alarm remote transmission telegram from the corresponding location number and sector is received. The flag is set until the alarms from the corresponding location number and sector is reset, or remote transmission inactive is received. The trouble flag is analogue to the fire flag as described above.
The parameter RTCfault uses fault remote transmission flag, and the RTCalarm the alarm remote transmission flag. The output of MF_RTC is static and follows the flags for remote transmission directly.
The Adf1Min and Adf1Max describes the sections for the possible alarms or troubles to be received. The remote transmission flag itself is only dependent on unit number and sector, but the section information is required to create the internal process image (and counters).

Symbol:



Syntax:

```
instruction :
MF_RTC ( 5: UnitAdrIn, Sector,
           Adf1Min, Adf1Max, RtcEvent);
         1: DO);
```

Parameters:

UnitAdrIn	Location	100, 110..118,
.		120..128,...., 240..248,
.		800..862, 900.
Sect	Sector	A, B, C, D, E, F, 0.
ADF1Min	Address field 1 min.	00..FD.
ADF1Max	Address field 1 max.	00..FD.
RtcEvent	Remote tans. Event	RTCalarm, RTCfault.
DO	Link-ID of DO	OutSignal.

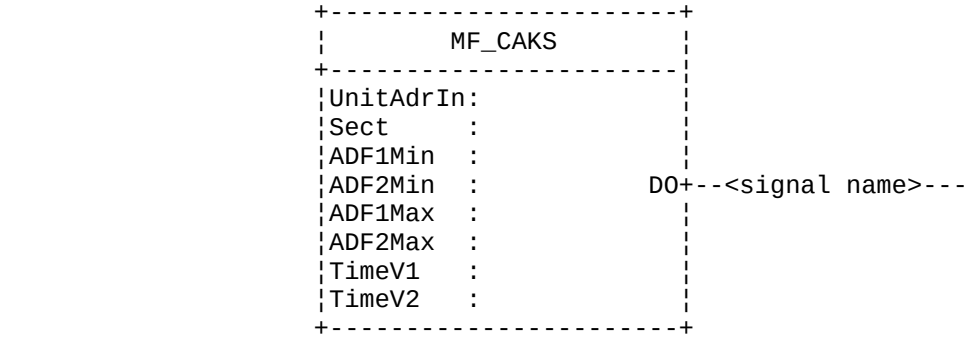
Example:

```
a_mf_rtc :MF_RTC ( 5: 111, B, B1, B8, RTCalarm; 1: M_DO_1_1_S);
```

12.3.11 MF_CAKS

Description:
MF_CAKS is an element specially dedicated to '**C**erberus **A**larm **K**oncept'. When a alarm telegram is received two timers are started (V1 and V2). An acknowledge command stops V1 and resetting the alarm will stop V2. If V1 expires before the alarm is acknowledged, or V2 before the alarm is reset, the output of the element is activated (V1 or V2 expired). The description above is only valid when the control unit is in DAY organisation. If the control unit is set to NIGHT organisation, V1/V2 is bypassed (V1 = V2 = 0).
The output is immediately activated if more than one automatic detector signals an alarm, or if a manual call point is activated. The output of the element is static.

Symbol:



Syntax:

```
instruction :
MF_CAKS (  8: UnitAdrIn, Sect, ADF1Min, ADF2Min,
              ADF1Max, ADF2Max, TimeV1, TimeV2;
           1: DO
```

Parameters:

UnitAdrIn	Location	100, 110..118, 120..128,..., 240..248, 800..862, 900.
Sect	Sector	A, B, C, D, E, F, 0.
ADF1Min	Address field 1 min.	00..FD.
ADF2Min	Address field 2 min.	00..FD.
ADF1Max	Address field 1 max.	00..FD.
ADF2Max	Address field 2 max.	00..FD.
TimeV1		0..999 seconds
TimeV2		0..999 seconds
DO	Link-ID of DO	OutSignal.

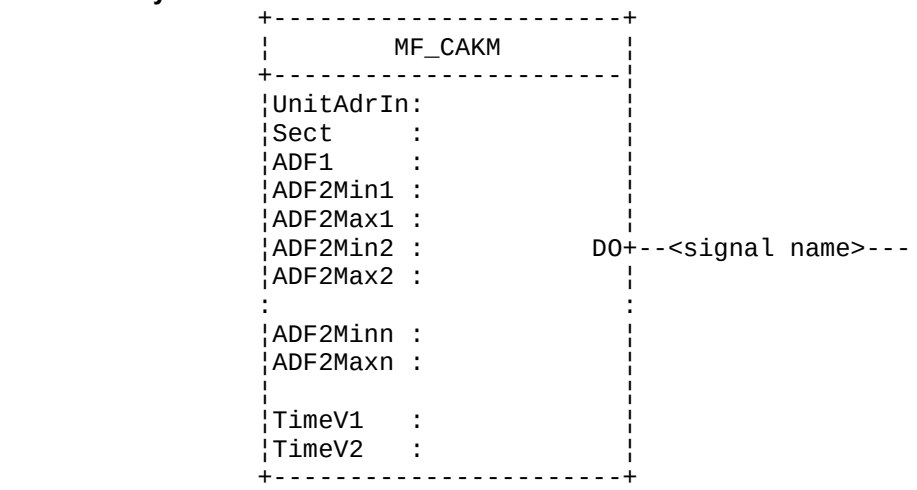
Example:

```
acaks_1 :      MF_CAKS ( 8: 111, A, 01, 35: 05, 50, 60, 360;
                         1: M_DO_1_5_S);
```

12.3.12 MF_CAKM

Description:
MF_CAKM is almost identical to MF_CAKS, the only difference being that MF_CAKM may have several data ranges of ADF2 (up to 100 ranges).

Symbol:



Syntax:

```
instruction :
MF_CAKS (  3: UnitAdrIn, Sect, ADF1;
         2*n: ADF2Min1, ADF2Max1,
              ADF2Min1, ADF2Max1, ...,
              ADF2Minn, ADF2Maxn;,
           2: TimeV1, TimeV2;
           1: DO);
```

Parameters:

UnitAdrIn	Location	100, 110..118, 120..128,..., 240..248, 800..862, 900.
Sect	Sector	A, B, C, D, E, F, 0.
ADF1	Address field 1 min.	00..FD.
ADF2Min1..n	Address field 2 min.	00..FD.
ADF2Max1..n	Address field 2 max.	00..FD.
TimeV1		0..999 seconds
TimeV2		0..999 seconds
DO	Link-ID of DO	OutSignal.

Example:

```
acaks_1 :MF_CAKS ( 3: 111, A, A1;
                   4: 01, 04, 11, 17;
                   2: 60, 360;
                   1: M_DO_1_5_S);
```

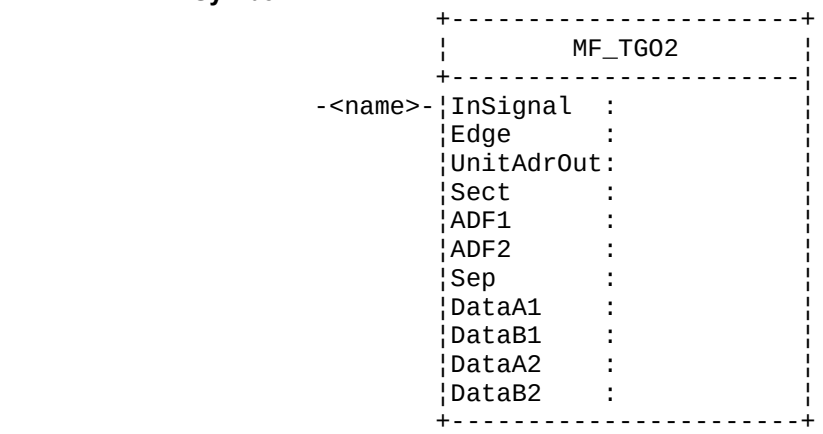
12.4 Telegram Output Elements

A telegram configured with data block A = 00 and data block B = 00 will not be transmitted. Though it is a syntactical allowed configuration. The usage is to send out only one telegram with the TGO-elements.

12.4.1 MF_TGO2

Description:
TGO2 is the standard output element for transmitting telegrams to CERLOOP/CERBAN. Its input is a signal name. This can be an invented name from a logic element or a TGI, or it can be a digital input (D_IN). The behaviour of the element depends on the input parameter 'Edge'.

Symbol:



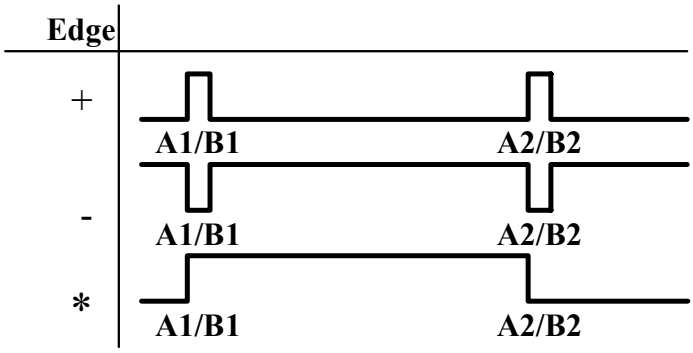
Syntax:

```
instruction :
MF_TGO2 (1: InSignal; 1: Edge;
                      9: UnitAdrOut, Sect, ADF1, ADF2, Sep,
                         DataA1, DataB1, DataA2, DataB2);
```

Parameters:

InSignal	Link-ID of input	InSignal.
Edge		+, -, *.
UnitAdrOut	Location	000, 100, 110..118, 120..128,..., 240..248, 800..862, 900.
Sect	Sector	A, B, C, D, E, F, 0.
ADF1	Address field 1	00..FD.
ADF2	Address field 2	00..FD.
Sep	Separator	M, R, N, U, Q.
DataA1	Data block A1	00..FD.
DataB1	Data block B1	00..FD.
DataA2	Data block A2	00..FD.
DataB2	Data block B2	00..FD.

The figure below shows how data blocks A1, B1 and A2, B2 are transmitted.



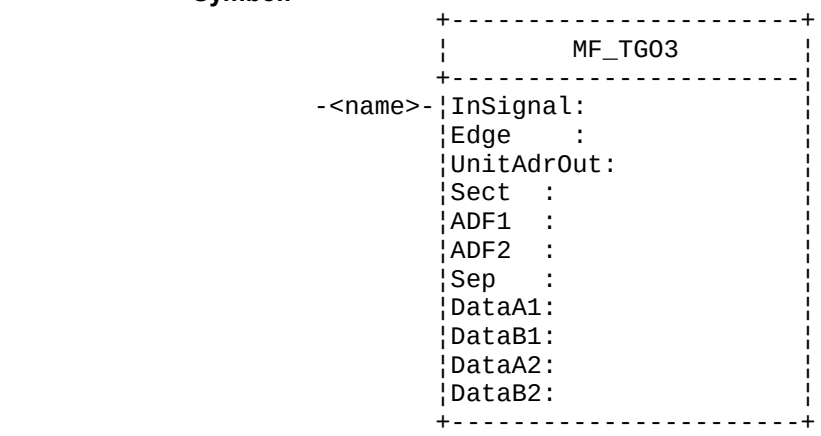
Example :

```
a_tgo2_1 :MF_TGO2 ( 1: in0001; 1: *;
                    9: 141, E, E1, 51, R, 61, 4F, 61, 4D);
```

12.4.2 MF_TGO3

Description:
TGO3 is an output element for transmitting a telegram to itself. This telegram is not visible on the network. Its input is a signal name. This can be an invented name from a logic element or a TGI, or it can be a digital input (D_IN). The behaviour of the element depends on the input parameter 'Edge' (see description TGI2).

Symbol:



Syntax:

```
instruction:
MF_TGO3 (1: InSignal; 1: Edge;
                      9: UnitAdrOut, Sect, ADF1, ADF2, Sep,
                         DataA1, DataB1, DataA2, DataB2);
```

Parameters:

InSignal	Link-ID of input	InSignal.
Edge		+, -, *.
UnitAdrOut	Location	000, 100, 110..118, 120..128,..., 240..248, 800..862, 900.
Sect	Sector	A, B, C, D, E, F, 0.
ADF1	Address field 1	00..FD.
ADF2	Address field 2	00..FD.
Sep	Separator	M, R, N, U, Q.
DataA1	Data block A1	00..FD.
DataB1	Data block B1	00..FD.
DataA2	Data block A2	00..FD.
DataB2	Data block B2	00..FD.

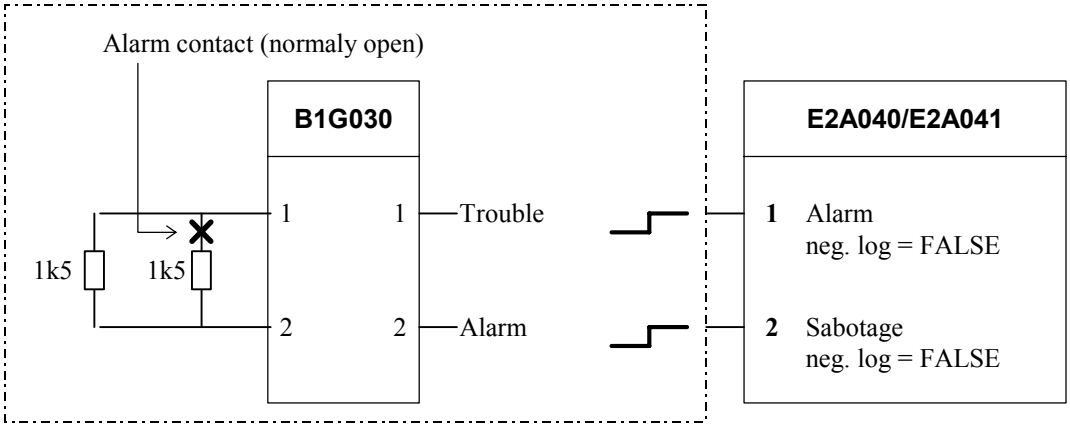
Example:

```
a_tgo3_1 :MF_TGO3 ( 1: in0001; 1: *;
                    9: 141, E, E1, 51, R, 61, 4F, 61, 4D);
```

13 Element description - MM7033 Elements

State-event diagrams of these elements can be found in Appendix B. See AH0.1 for defined DMS7000 telegrams.

13.1 Usage of Line Monitoring for MM Elements (B1G030)



B1G030 in normal use is with the alarm contact open. The alarm output of the B1G030 goes high (TRUE) when the contact closes. The trouble output goes high if a short circuit or an open circuit of the inputs of B1G030 occur.

"Truth table" of B1G030 in normal use:

	1	2	
1k5	0	0	= NORMAL OPERATION
750ohm	0	1	= ALARM
open c	1	0	= TROUBLE
short c	1	0	= TROUBLE

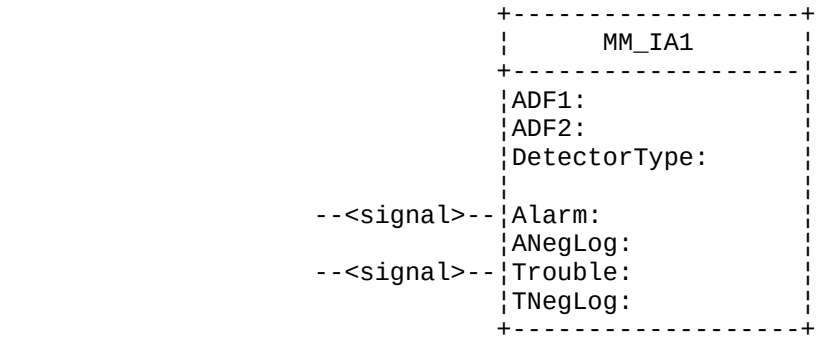
B1G030 is not designed to be used with the alarm contact normally closed.

13.2 MM_IA1

Description:
The basis for this element is a CZ10 with collective lines. The element requires two MUX inputs, as the alarm input is monitored by a B1G030. Trouble signals have higher priority than alarms. Negative logic (input in normal state) can be defined independently for the alarm input and trouble input.

Limitations:
- does not distinguish between a local alarm and a general alarm.

Symbol:



Syntax:

```
instruction :
MM_IA1 ( 5: ADF1, ADF2, DetectorType, ANegLog, TNegLog;
         2: Alarm, Trouble);
```

Parameters:

ADF1	Address field 1	A1..A9.
ADF2	Address field 2	01..99.
DetectorType		general, automatic, manual.
ANegLog	Negative logic alarm signal	TRUE, FALSE.
TNegLog	Negative logic trouble signal	TRUE, FALSE.
Alarm	Link-ID - alarm	InSignal.
Trouble	Link-ID - trouble	InSignal.

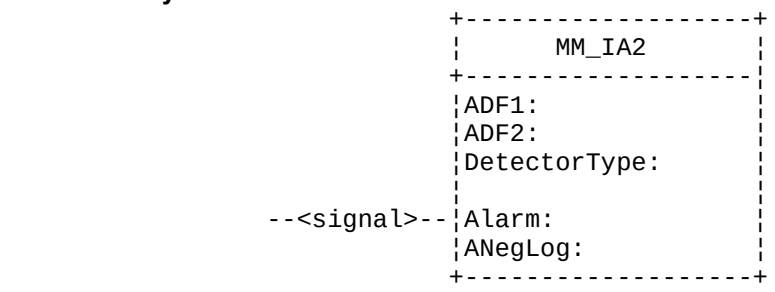
Example:

```
a_mm_ia1 :MM_IA1 ( 5: A1, 01, automatic, FALSE, FALSE;
                   2: M_DI_1_1_S, M_DI_1_2_S);
```

13.3 MM_IA2

Description:
This element is equivalent to the MM_IA1, but without the trouble signal. See MM_IA1 for further information.

Symbol:



Syntax:

```
instruction :
MM_IA2 ( 4: ADF1, ADF2, DetectorType, ANegLog;
         1: Alarm);
```

Parameters:

ADF1	Address field 1	A1..A9.
ADF2	Address field 2	01..99.
DetectorType		general, automatic, manual.
ANegLog	Negative logic alarm signal	TRUE, FALSE.
Alarm	Link-ID - alarm	InSignal.

Example:

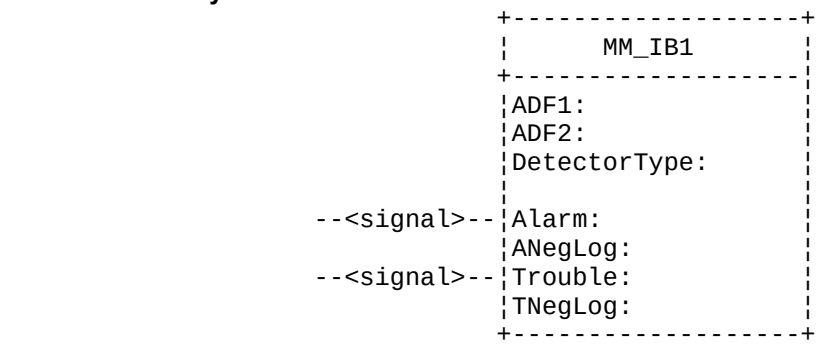
```
a_mm_ia2 :MM_IA2 ( 4: A1, 01, automatic, FALSE; 1: M_DI_1_1_S);
```

13.4 MM_IB1

Description:
The basis for this element is a CZ10 with collective lines. The element require two MUX inputs, as the alarm input is monitored by the B1G030. Trouble signals have higher priority than alarms. Negative logic (input in normal state) can be defined independantly for the alarm input and trouble input.

Limitations:
- does not distinguish between a local alarm and a general alarm
- no test function

Symbol:



Syntax:

```
instruction :
MM_IB1 ( 5: ADF1, ADF2, DetectorType, ANegLog, TNegLog;
         2: Alarm, Trouble);
```

Parameters:

ADF1	Address field 1	B1..B9.
ADF2	Address field 2	01..99.
DetectorType		general, automatic, manual.
ANegLog	Negative logic alarm signal	TRUE, FALSE.
TNegLog	Negative logic trouble signal	TRUE, FALSE.
Alarm	Link-ID - alarm	InSignal.
Trouble	Link-ID - trouble	InSignal.

Example:

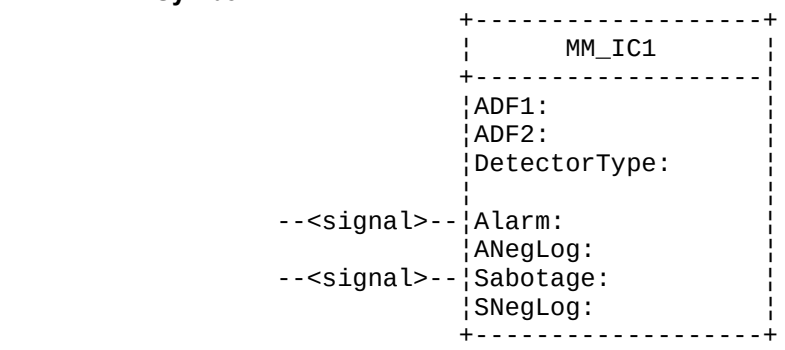
```
a_mm_ib1 :MM_IB1 ( 5: B1, 01, automatic, FALSE, FALSE;
                   2: M_DI_1_1_S, M_DI_1_2_S);
```

13.5 MM_IC1

Description:
The basis for this element is a CZ12. See also MM_IC1a. It requires two MUX inputs, as the alarm input is monitored by a B1G030. Negative logic (input in normal state) can be defined independently for alarm input and trouble input.

Limitations:
- does not distinguish between a local alarm and a general alarm
- test alarm for alarm, alarmsabo and sabo send the same telegram.

Symbol:



Syntax:

```
instruction :
MM_IC1 ( 5: ADF1, ADF2, DetectorType, ANegLog,SNegLog;¦
         2: Alarm, Sabotage);
```

Parameters:

ADF1	Address field 1	C1..C9.
ADF2	Address field 2	01..99.
DetectorType		general, burglary, holdup, theft.
ANegLog	Negative logic alarm signal	TRUE, FALSE.
SNegLog	Negative logic sabotage signal	TRUE, FALSE.
Alarm	Link-ID - alarm	InSignal.
Sabotage	Link-ID - sabotage	InSignal.

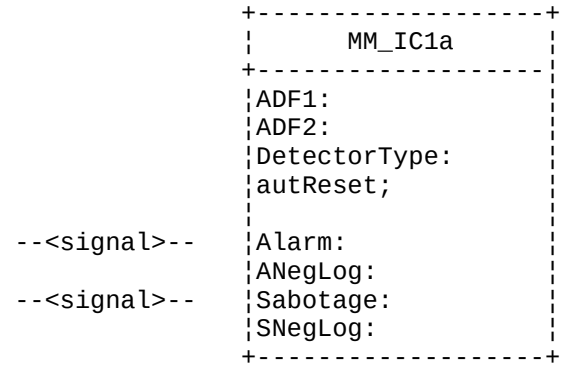
Example:

```
a_mm_ic1 :MM_IC1 ( 5: C1, 01, theft, FALSE, FALSE;
                   2: M_DI_1_1_S, M_DI_1_2_S);
```

13.6 MM_IC1a

Description:
This element is equivalent to the MM_IC1. In addition to the standard MM_IC1 there is an optional automatic reset after timeout included. See MM_IC1 for further information.

Symbol:



Syntax:

```
instruction :
MM_IC1a( 6: ADF1, ADF2, DetectorType, ANegLog, SNegLog,
            autReset;
         2: Alarm, Sabotage);
```

Parameters:

ADF1	Address field 1	C1..C9.
ADF2	Address field 2	01..99.
DetectorType		general, burglary, holdup, theft.
ANegLog	Negative logic alarm signal	TRUE, FALSE.
SNegLog	Negative logic sabotage signal	TRUE, FALSE.
Alarm	Link-ID - alarm	InSignal.
Sabotage	Link-ID - sabotage	InSignal.
AutReset	automatic Reset after timeout	*, 0..86400 sec

. * = no automatic Reset

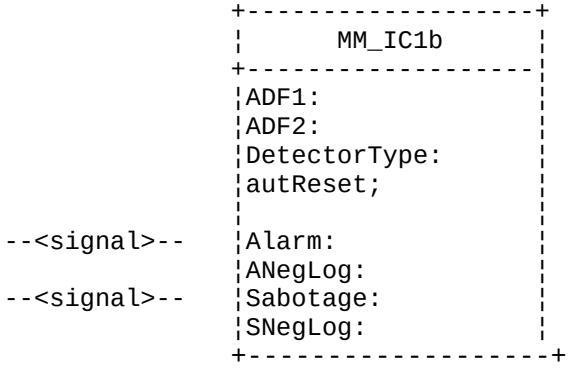
Example:

```
a_mm_ic1a :MM_IC1a ( 6: C1, 01, theft, FALSE, FALSE, 180;
                     2: M_DI_1_1_S, M_DI_1_2_S);
```

13.7 MM_IC1b

Description:
This element is equivalent to the MM_IC1, but without the OFF-command (state). In addition to the standard MM_IC1 there is an optional automatic reset after timeout included. See MM_IC1 for further information.

Symbol:



Syntax:

```
instruction :
MM_IC1b( 6: ADF1, ADF2, DetectorType, ANegLog, SNegLog,
            autReset;
         2: Alarm, Sabotage);
```

Parameters:

ADF1	Address field 1	C1..C9.
ADF2	Address field 2	01..99.
DetectorType		general, burglary, holdup, theft.
ANegLog	Negative logic alarm signal	TRUE, FALSE.
SNegLog	Negative logic sabotage signal	TRUE, FALSE.
Alarm	Link-ID - alarm	InSignal.
Sabotage	Link-ID - sabotage	InSignal.
AutReset	automatic Reset after timeout	*, 0..86400 seconds. * = no automatic Reset

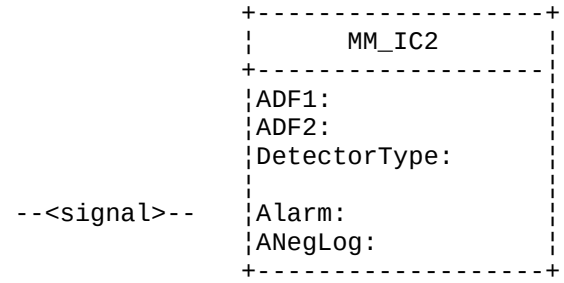
Example:

```
a_mm_ic1b :MM_IC1b ( 6: C1, 01, theft, FALSE, FALSE, 180;
                    2: M_DI_1_1_S, M_DI_1_2_S);
```

13.8 MM_IC2

Description:
This element is equivalent to the MM_IC1, but without the sabotage signal. See MM_IC1 for further information.

Symbol:



Syntax:

```
instruction :
MM_IC2 ( 4: ADF1, ADF2, DetectorType, ANegLog;
         1: Alarm);
```

Parameters:

ADF1	Address field 1	C1..C9.
ADF2	Address field 2	01..99.
DetectorType		general, burglary, holdup, theft.
ANegLog	Negative logic alarm signal	TRUE, FALSE.
Alarm	Link-ID - alarm	InSignal.

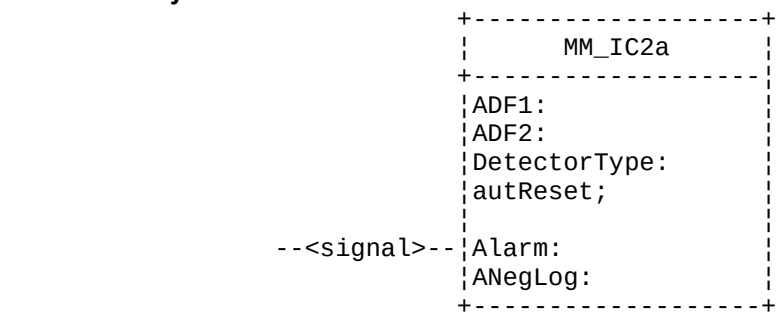
Example:

```
a_mm_ic2 :MM_IC2 ( 4: C1, 01, theft, FALSE; 1: M_DI_1_1_S);
```

13.9 MM_IC2a

Description:
This element is equivalent to the MM_IC1**a**, but without the sabotage signal. See MM_IC1**a** for further information.

Symbol:



Syntax:

```
instruction :
MM_IC2a( 5: ADF1, ADF2, DetectorType, ANegLog,autReset;
         1: Alarm);
```

Parameters:

ADF1	Address field 1	C1..C9.
ADF2	Address field 2	01..99.
DetectorType		general, burglary, holdup, theft.
ANegLog	Negative logic alarm signal	TRUE, FALSE.
AutReset	automatic Reset after timeout	*, 0..86400 sec. * = no automatic Reset
Alarm	Link-ID - alarm	InSignal.

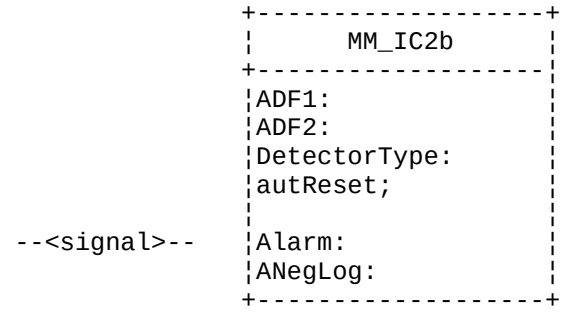
Example:

```
a_mm_ic2a :MM_IC2a ( 5: C1, 01, theft, FALSE, 180;
                     1: M_DI_1_1_S);
```

13.10 MM_IC2b

Description:
This element is equivalent to the MM_IC1**b**, but without the sabotage signal. See MM_IC1**b** for further information.

Symbol:



Syntax:

```
instruction :
MM_IC2b( 5: ADF1, ADF2, DetectorType, ANegLog,autReset;
         1: Alarm);
```

Parameters:

ADF1	Address field 1	C1..C9.
ADF2	Address field 2	01..99.
DetectorType		general, burglary, holdup, theft.
ANegLog	Negative logic alarm signal	TRUE, FALSE.
AutReset	automatic Reset after timeout	*, 0..86400 sec. * = no automatic Reset
Alarm	Link-ID - alarm	InSignal.

Example:

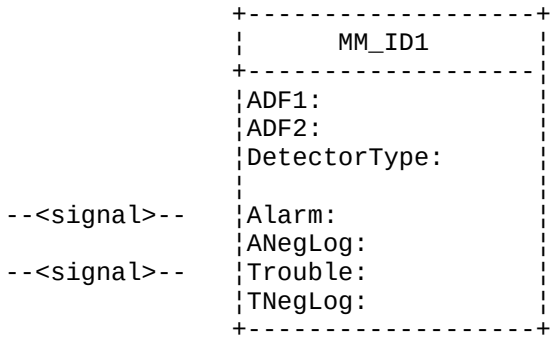
```
a_mm_ic2b :MM_IC2b ( 5: C1, 01, theft, FALSE, 180;
                     1: M_DI_1_1_S);
```

13.11 MM_ID1

Description:
This element requires two MUX inputs, as the alarm input is monitored by the B1G030. Trouble signals have a higher priority than alarms. Negative logic (input in normal state) can be defined independently for the alarm input and trouble input.

Limitations:
- does not distinguish between prewarning and gas-alarm
- no test function.

Symbol:



Syntax:

```
instruction :
MM_ID1 ( 5: ADF1, ADF2, DetectorType, ANegLog, TNegLog;
         2: Alarm, Trouble);
```

Parameters:

ADF1	Address field 1	D1..D9.
ADF2	Address field 2	01..99.
DetectorType		general, automatic, manual.
ANegLog	Negative logic alarm signal	TRUE, FALSE.
TNegLog	Negative logic trouble signal	TRUE, FALSE.
Alarm	Link-ID - alarm	InSignal.
Trouble	Link-ID - trouble	InSignal.

Example:

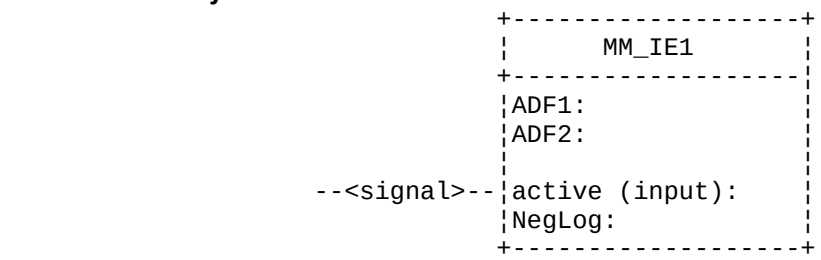
```
a_mm_id1 :MM_ID1 ( 5: D1, 01, manual, FALSE, FALSE;
                   2: M_DI_1_1_S, M_DI_1_2_S);
```

13.12 MM_IE1

Description:
This element corresponds to CZ10 sector E1. It requires only 1 MUX input, which means that monitoring is not possible. When the input is activated, a telegram with the dataA,B = 414F is transmitted. This message must be acknowledged before the message normal operation can be received. If the input goes inactive before the active message was acknowledged, the element remains in the active state (latched input).

- refer to the state event diagram for details.

Symbol:



Syntax:

```
instruction :
MM_IE1 ( 3: ADF1, ADF2, NegLog;
         1: Active);
```

Parameters:

ADF1	Address field 1	E1..E9.
ADF2	Address field 2	01..99.
NegLog	Negative logic signal	TRUE, FALSE.
Active	Link-ID - active	InSignal.

Example:

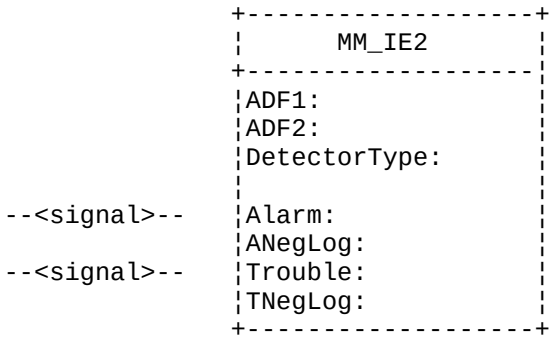
```
a_mm_ie1 :MM_IE1 ( 3: E1, 01, FALSE; 1: M_DI_1_1_S);
```

13.13 MM_IE2

Description:
The basis for this element is a CZ10 with sector E2 – building services. The element require two MUX inputs, as the alarm input is monitored by the B1G030. Trouble signals have a higher priority than alarms. Negative logic (input in normal state) can be defined independently for the alarm input and trouble input.

Limitations:
- does not distinguish between a local alarm and a general alarm

Symbol:



Syntax:

```
instruction :
MM_IE2 ( 5: ADF1, ADF2, DetectorType, ANegLog, TNegLog;
         2: Alarm, Trouble);
```

Parameters:

ADF1	Address field 1	E1..E9.
ADF2	Address field 2	01..99.
DetectorType		general, automatic, manual.
ANegLog	Negative logic alarm signal	TRUE, FALSE.
TNegLog	Negative logic trouble signal	TRUE, FALSE.
Alarm	Link-ID - alarm	InSignal.
Trouble	Link-ID - trouble	InSignal.

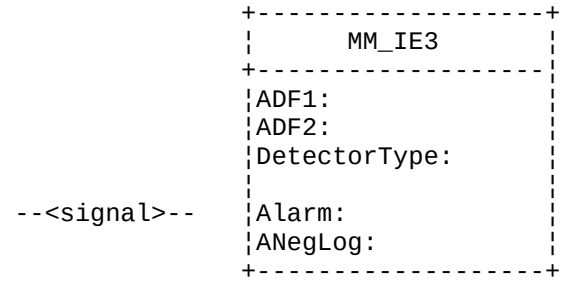
Example:

```
a_mm_ie2 :MM_IE2 ( 5: E1, 01, automatic, FALSE, FALSE;
                   2: M_DI_1_1_S, M_DI_1_2_S);
```

13.14 MM_IE3

Description:
This element is equivalent to the MM_IE2, but without the trouble signal. See MM_IE2 for further information.

Symbol:



Syntax:

```
instruction :
MM_IE3 ( 4: ADF1, ADF2, DetectorType, ANegLog;
         1: Alarm);
```

Parameters:

ADF1	Address field 1	E1..E9.
ADF2	Address field 2	01..99.
DetectorType		general, automatic, manual.
ANegLog	Negative logic alarm signal	TRUE, FALSE.
Alarm	Link-ID - alarm	InSignal.

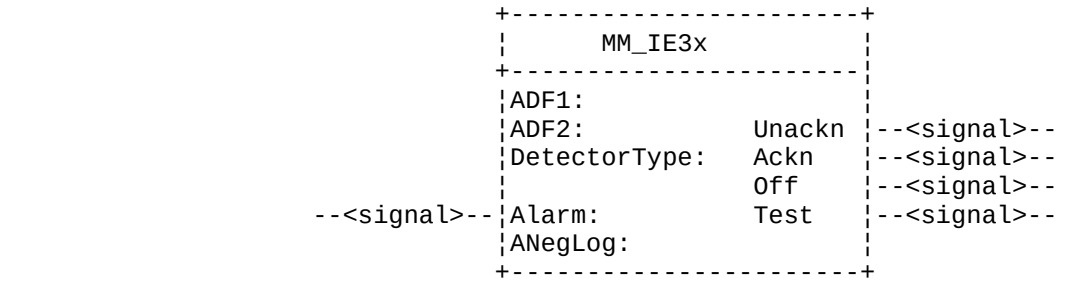
Example:

```
a_mm_ie3 :MM_IE3 ( 4: E2, 01, manual, FALSE; 1: M_DI_1_1_S);
```

13.15 MM_IE3x

Description:
This element is equivalent to the MM_IE3, but with additional 4 output signals.

Symbol:



Syntax:

```
instruction :
MM_IE3x (4: ADF1, ADF2, DetectorType, ANegLog;
         1: Alarm;
         4: Unackn, Ackn, Off, Test);
```

Parameters:

ADF1	Address field 1	E1..E9.
ADF2	Address field 2	01..99
DetectorType		general, automatic, manual.
ANegLog	Negative logic alarm signal	TRUE, FALSE
Alarm	Link-ID - alarm	InSignal.
Unackn	Link-ID of Unackn	OutSignal.
Ackn	Link-ID of Ackn	OutSignal.
Off	Link-ID of Off	OutSignal.
Test	Link-ID of Test	OutSignal.

Example:

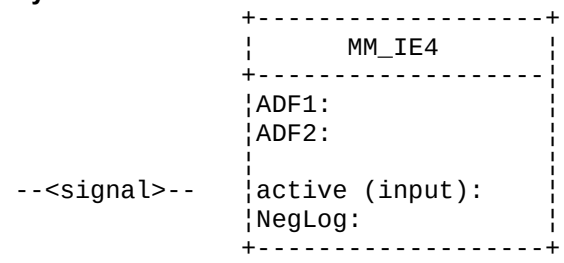
```
a_mm_ie3x :MM_IE3x (4: E2, 01, manual, FALSE;
                    1: M_DI_1_1_S;
                    4: M_DO_1_1_S, M_DO_1_2_S, M_DO_1_3_S,
                       M_DO_1_4_S);
```

13.16 MM_IE4

Description:
This element corresponds to CZ10 sector E1. It requires only 1 MUX input, which means that monitoring is not possible. When the input is activated, a telegram with the dataA,B = 414F is transmitted. Whenever the input goes inactive, the telegram with the dataA,B = 413C is transmitted (independent of acknowledgement of the telegram)

- refer to the state event diagram for details.

Symbol:



Syntax:

```
instruction :
MM_IE4 ( 3: ADF1, ADF2, NegLog;
         1: Active);
```

Parameters:

ADF1	Address field 1	E1..E9.
ADF2	Address field 2	01..99.
NegLog	Negative logic signal	TRUE, FALSE.
Active	Link-ID - active	InSignal.

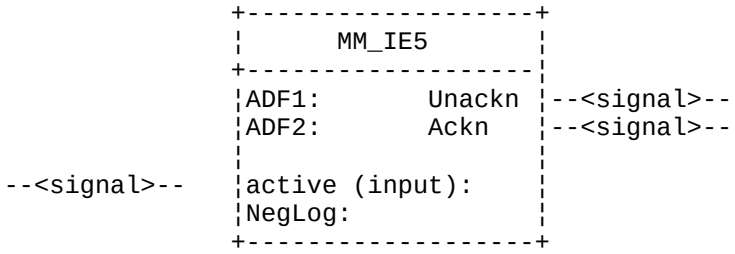
Example:

```
a_mm_ie4 :MM_IE4 ( 3: E1, 01, FALSE; 1: M_DI_1_1_S);
```

13.17 MM_IE5

Description:
This element is equivalent to the MM_IE4, but with additional 2 output signals. Corresponding telegram to send for "activ low" is new 414D. (MM_IE4 = 413C)

Symbol:



Syntax:

```
instruction :
MM_IE5 ( 3: ADF1, ADF2, NegLog;
         1: Active;
         2: Unackn, Ackn);
```

Parameters:

ADF1	Address field 1	E1..E9.
ADF2	Address field 2	01..99.
NegLog	Negative logic signal	TRUE, FALSE.
Active	Link-ID - active	InSignal.
Unackn	Link-ID of Unackn	OutSignal.
Ackn	Link-ID of Ackn	OutSignal.

Example:

```
a_mm_ie5 :MM_IE5 ( 3: E1, 01, FALSE;
                   1: M_DI_1_1_S;
                   2: M_DO_1_1_S, M_DO_1_2_S);
```

13.18 MM_OE1

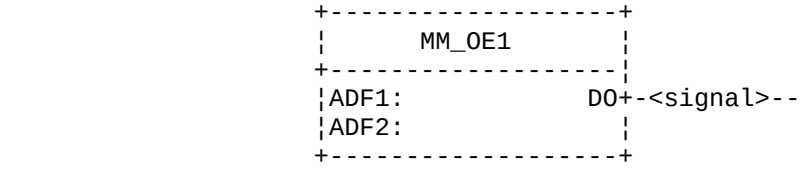
Description:
The basis for this element is a CZ10 with a building services sector. An output can be remotely controlled. In addition, it is also possible to block this output in the active or inactive state.
This is done as follows:
8. initial state is output not blocked and inactive.
- output can be activated with the normal telegram R614F
- MM_OE1 is able to block the output in the inactive state by sending the telegram Rxx56
- unblocking the output is done with telegram Rxx55.

9. initial state is output blocked and inactive
- it is not possible to activate the output when is already blocked in the inactive mode; it must first be unblocked (telegram Rxx55)
- the output can now be activated (with telegram R614F)
- it can be blocked in the active mode (telegram Rxx56)
- unblocking is again done with telegram Rxx55.

10. initial state is output blocked and active
- it is not possible to turn the output off when it is 'isolated' in the on mode; it must first be unblocked (telegram Rxx55)
- the output can now be turned off with telegram R614D.

xx = don't care

Symbol:



Syntax:

```
instruction :
MM_OE1 ( 2: ADF1, ADF2;
         1: DO);
```

Parameters:

ADF1	Address field 1	E1..E9.
ADF2	Address field 2	01..99.
DO	Link-ID - DO	OutSignal.

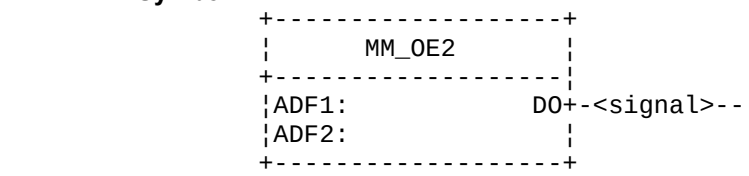
Example:

```
a_mm_oe1 :MM_OE1 ( 2: E1, 01;
                   1: M_DO_1_1_S);
```

13.19 MM_OE2

Description:
The characteristics of this element are similar to MM_OE1, the only difference being that the output cannot be blocked in the active state. When the actual state is 'output on - not isolated' (unblocked) and a isolate teleram is received (Rxx56) the new state is 'output off - isolated'. See MM_OE1 for further details.

Symbol:



Syntax:

```
instruction :
MM_OE2 ( 2: ADF1, ADF2;
         1: DO);
```

Parameters:

ADF1	Address field 1	E1..E9.
ADF2	Address field 2	01..99.
DO	Link-ID - DO	OutSignal.

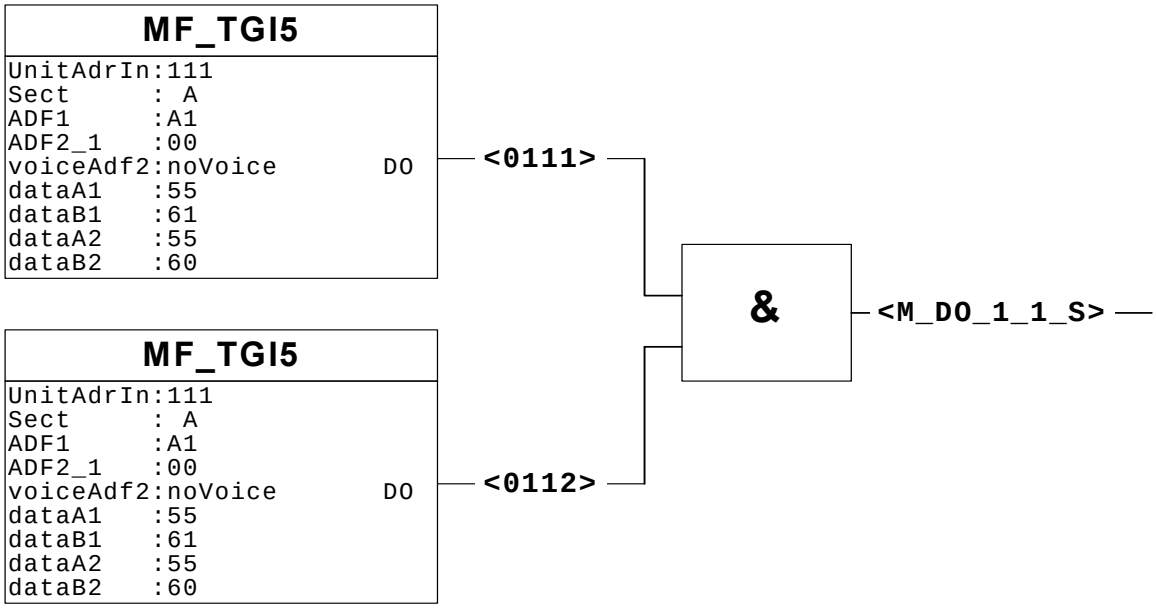
Example:

```
a_mm_oe2 :MM_OE2 ( 2: E1, 01;
                   1: out0001);
```

14 Examples

Several examples of PLC file configurations are shown below. The elements and their associated logic are shown first and this is followed by the required code lines in the PLC file. Examples 1.xx and 2.xx are MF examples. MM examples are included in the element descriptions (MM tables are mostly monotonous).

Example 1.01:
Organization monitoring of fire control units with location numbers 111 - 112. The output is active when all control units have the same organisation.

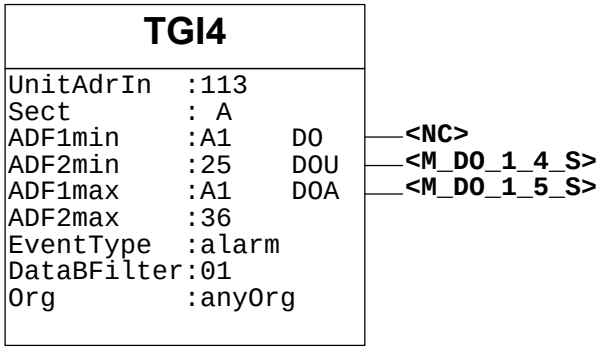


Example 1.02:
The LED connected to DEMUX card number 1, line 1 should be lit when:

- an unacknowledged alarm is present from any automatic detector in zones A125 to A136 of the control unit with location number 113.

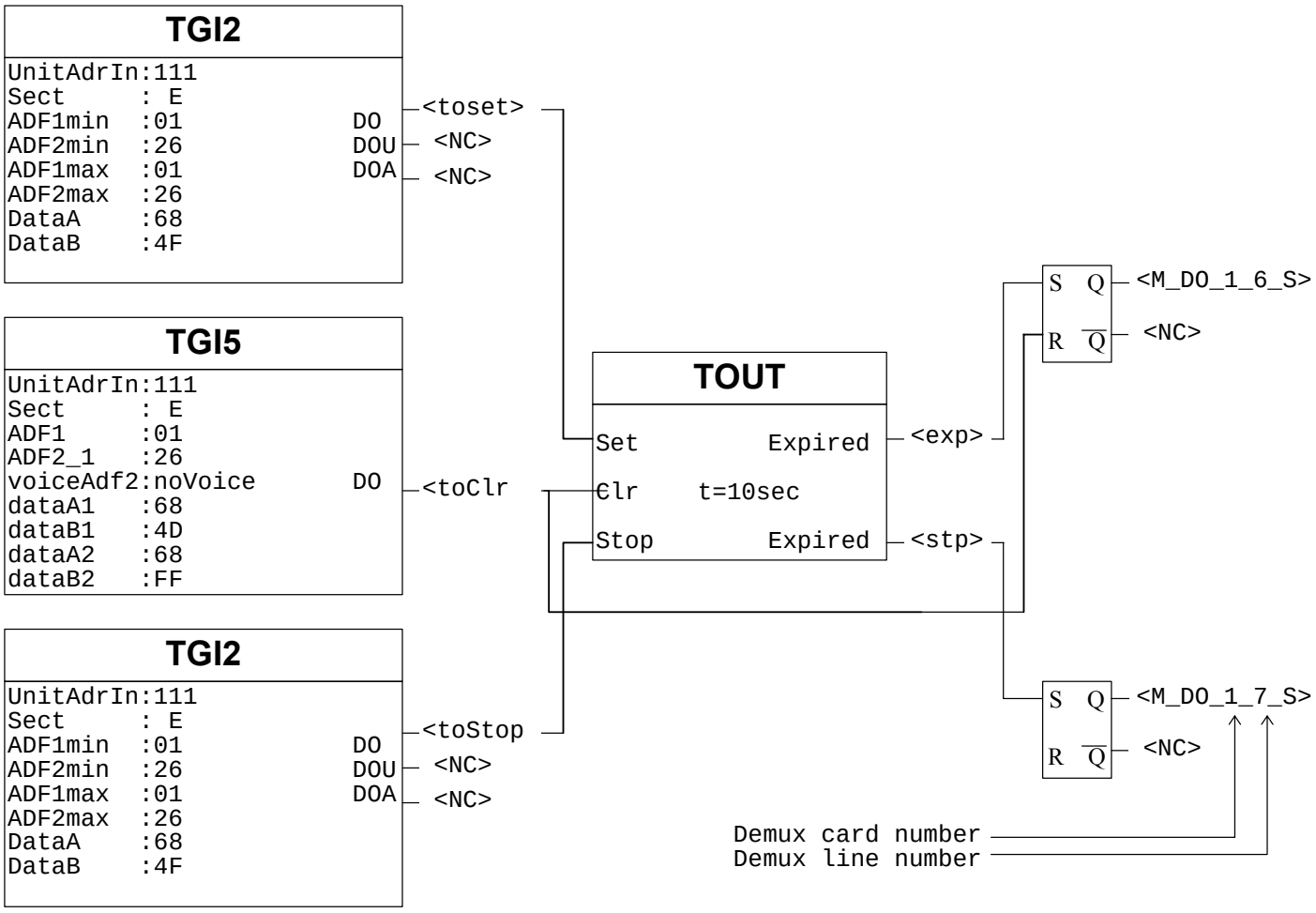
The LED connected to DEMUX card number 1, line 2 should be lit when:

- an acknowledged alarm is present from any automatic detector in zones A125 to A136 of the control unit with location number 113.



Example 1.03:

Application of E90Ci with confirm. When the control element number 26 on line 01 is activated, it must be confirmed within 10 seconds. A confirm within 10s will activate the output stopped. No confirm (or confirm after 10s) will activate the expired output.



This is the complete file of the examples 1.xx. The configuration corresponds to the recommended structure of the PLC file.

```
HEADER
  Project............: MF7033
  Purpose............: Example of PLC syntax...
  Installation.......: MF_DOC.Q00
  ON-Nr..............:
  Author.............: Hge
  Creation date......: 25.06.92

  Modification remark: Corrected structure of the PLC-file
  Modified by........: Hge
  Modification date..: 4-AUG-92

  Modification remark: ---
  Modified by........: ---
  Modification date..: ---
END

SEGMENT Digital_Inputs
  (* no input elements *)
END Digital_Inputs

SEGMENT Pre_Logic_FL
  (* No logic before MF/MM elements *)
END Pre_Logic_FL

SEGMENT MF_Functions_FL
  e90Ci_active_to_set_on_tout
  : MF_TGI2 ( 9: 113, E, 1, 26, 1, 26, 68, 4F, anyOrg;
              3: toSet, NC, NC);
  e90Ci_confirm_to_stop_on_tout
  : MF_TGI2 ( 9: 113, E, 1, 26, 1, 26, 68, 63, anyOrg;
              3: toStop, NC, NC);
  alarm_aut_det
  : MF_TGI4 ( 9: 113, A, A1, 25, A1, 36, alarm, 01, anyOrg;
              3: NC, M_DO_1_4_S, M_DO_1_5_S);
  org_location_111
  : MF_TGI5 ( 3: 111, A, A1; 1: 00; 5: noVoice, 55, 61, 55, 60;
              1: o111);
  org_location_112
  : MF_TGI5 ( 3: 112, A, A1; 1: 00; 5: noVoice, 55, 61, 55, 60;
              1: o112);
  e90Ci_inactive_to_clr_on_tout
  : MF_TGI5 ( 3: 113, E, 1; 1: 26; 5: noVoice, 68, 4D, 68, FF;
              1: toClr);
END MF_Functions_FL
SEGMENT Post_Logic_FL
  org_output_day : F_AND ( 2: o111, o112; 1: M_DO_1_1_S);
  e90ci_timeout  : F_TOUT ( 1: 60; 3: toSet, toClr, toStop; 2: exp, stp);
  e90Ci_expired  : F_LATCH ( 2: exp, toClr; 2: M_DO_1_6_S, NC);
  e90Ci_stopped  : F_LATCH ( 2: stp, toClr; 2: M_DO_1_7_S, NC);
END Post_Logic_FL

(* DEMUX output elements *)
SEGMENT Digital_Outputs
  org_output_day  : D_OUT (2: M_DO_1_1_S, K_FALSE; 2: posLogic, noLEDtest);
  alarm_aut_det_U : D_OUT (2: M_DO_1_4_S, K_FALSE; 2: posLogic, noLEDtest);
  alarm_aut_det_A : D_OUT (2: M_DO_1_5_S, K_FALSE; 2: posLogic, noLEDtest);
  e90Ci_expired   : D_OUT (2: M_DO_1_6_S, K_FALSE; 2: posLogic, noLEDtest);
  e90Ci_stopped   : D_OUT (2: M_DO_1_7_S, K_FALSE; 2: posLogic, noLEDtest);
END Digital_Outputs
```

Example 2.00

The example below is a realistic application for the use of a PIA module. A relay card E3G020 is connected to the PIA port P2. PIA port P1 is always used for power supply monitoring. The following points are evaluated in the configuration file below:

Relay 1: Active on all alarms, organisation night, sector fire, only the fire control units 111..117, no difference unacknowledged / acknowledged.

Relay 2: Active on all troubles, all sectors (A, 0), from the control units 111..117. no difference unacknowledged/acknowledged.

Relay 3: Active on alarm remote transmission (094F) from control unit 121 (CZ12).

Relay 4: Active on alarm remote transmission (094F) from control unit 118 (CZ12).

Relay 5: Active when control unit 118 or 121 is off line.

```
HEADER
  Project............: MF7033
  Purpose............: Example 2, Outputs to PIA (E3G020)
  Installation.......: PIATEST.Q00
  ON-Nr..............:
  Author.............: Hge
  Creation date......: 29.06.92

  Modification remark: Corrected structure of the PLC-file
  Modified by........: Hge
  Modification date..: 4-AUG-92

END

SEGMENT Digital_Inputs
  (* no input elements *)
END Digital_Inputs

SEGMENT Pre_Logic_FL
  (* No logic before MF/MM elements *)
END Pre_Logic_FL

SEGMENT MF_Functions_FL
  (* Relay 1 active on all alarms, organization night, sector fire, control *)
  (* units 111 - 117.                                                      *)
  alarm_111
  : MF_TGI4 ( 9: 111, A, A1, 01, A1, 96, alarm, FF, night;
              3: a111, NC, NC);
  alarm_112
  : MF_TGI4 ( 9: 112, A, A1, 01, A1, 96, alarm, FF, night;
              3: a112,NC, NC);
  alarm_113
  : MF_TGI4 ( 9: 113, A, A1, 01, A1, 96, alarm, FF, night;
              3: a113, NC, NC);
  alarm_114
  : MF_TGI4 ( 9: 114, A, A1, 01, A1, 96, alarm, FF, night;
              3: a114, NC, NC);
  alarm_115
  : MF_TGI4 ( 9: 115, A, A1, 01, A1, 96, alarm, FF, night;
              3: a115, NC, NC);
  alarm_116
  : MF_TGI4 ( 9: 116, A, A1, 01, A1, 96, alarm, FF, night;
              3: a116, NC, NC);
  alarm_117
  : MF_TGI4 ( 9: 117, A, A1, 01, A1, 96, alarm, FF, night;
              3: a117, NC, NC);

  (* Relay 2 active on all troubles, sector FIRE, control units 111 - 117. *)
  fault_111_A
  : MF_TGI4 ( 9: 111, A, A1, ED, A1, ED, fault, FF, anyOrg;
              3: f111a, NC, NC);
  fault_112_A
  : MF_TGI4 ( 9: 112, A, A1, ED, A1, ED, fault, FF, anyOrg;
              3: f112a, NC, NC);
  fault_113_A
  : MF_TGI4 ( 9: 113, A, A1, ED, A1, ED, fault, FF, anyOrg;
              3: f113a, NC, NC);
```

```
  fault_114_A
  : MF_TGI4 ( 9: 114, A, A1, ED, A1, ED, fault, FF, anyOrg;
              3: f114a, NC, NC);
  fault_115_A
  : MF_TGI4 ( 9: 115, A, A1, ED, A1, ED, fault, FF, anyOrg;
              3: f115a, NC, NC);
  fault_116_A
  : MF_TGI4 ( 9: 116, A, A1, ED, A1, ED, fault, FF, anyOrg;
              3: f116a, NC, NC);
  fault_117_A
  : MF_TGI4 ( 9: 117, A, A1, ED, A1, ED, fault, FF, anyOrg;
              3: f117a, NC, NC);

  (* Relay 2 active on all troubles, sector BASIC, control units 111 - 117. *)
  fault_111_0
  : MF_TGI4 ( 9: 111, 0, 00, 00, 00, 0F, fault, FF, anyOrg;
              3: f1110, NC, NC);
  fault_112_0
  : MF_TGI4 ( 9: 112, 0, 00, 00, 00, 0F, fault, FF, anyOrg;
              3: f1120, NC, NC);
  fault_113_0
  : MF_TGI4 ( 9: 113, 0, 00, 00, 00, 0F, fault, FF, anyOrg;
              3: f1130, NC, NC);
  fault_114_0
  : MF_TGI4 ( 9: 114, 0, 00, 00, 00, 0F, fault, FF, anyOrg;
              3: f1140, NC, NC);
  fault_115_0
  : MF_TGI4 ( 9: 115, 0, 00, 00, 00, 0F, fault, FF, anyOrg;
              3: f1150, NC, NC);
  fault_116_0
  : MF_TGI4 ( 9: 116, 0, 00, 00, 00, 0F, fault, FF, anyOrg;
              3: f1160, NC, NC);
  fault_117_0
  : MF_TGI4 ( 9: 117, 0, 00, 00, 00, 0F, fault, FF, anyOrg;
              3: f1170, NC, NC);
(* Relay 3 active on Alarm Remote Transmission,sector C,control unit 121. *)
  fue_121 : MF_TGI5 ( 3: 121, C, 00; 1: EA;
                      5: noVoice, 09, 4F, 09, 4D; 1: M_PIAO_2_3_S);
  fue_118 : MF_TGI5 ( 3: 118, C, 00; 1: EA;
                      5: noVoice, 09, 4F, 09, 4D; 1: M_PIAO_2_4_S);

  (* Relay 5 active when control units 118, 121 not online *)
  online_118 : MF_ONLINE ( 2: 118, 70; 1: o118);
  online_121 : MF_ONLINE ( 2: 121, 70; 1: o121);
END MF_Functions


SEGMENT Post_Logic_FL
  alarmA  : F_OR ( 7: a111, a112, a113, a114, a115, a116, a117;
                   1: M_PIAO_2_1_S);
  faultA0 : F_OR ( 14: f111a, f112a, f113a, f114a, f115a, f116a, f117a,
                       f1110, f1120, f1130, f1140, f1150, f1160, f1170;
                    1: M_PIAO_2_2_S);
  online : F_OR ( 2: o118, o121; 1: M_PIAO_2_5_S);
END Post_Logic_FL

(* DEMUX output elements *)
SEGMENT Digital_Outputs
  (* Relay card E3G020 have negative logic *)
  alarmA  : D_OUT (2: M_PIAO_2_1_S, K_FALSE; 2: negLogic, noLEDtest);
  faultAC : D_OUT (2: M_PIAO_2_2_S, K_FALSE; 2: negLogic, noLEDtest);
  fue_121 : D_OUT (2: M_PIAO_2_3_S, K_FALSE; 2: negLogic, noLEDtest);
  fue_118 : D_OUT (2: M_PIAO_2_4_S, K_FALSE; 2: negLogic, noLEDtest);
  (* ONLINE is active when the unit is online *)
  online  : D_OUT (2: M_PIAO_2_5_S, K_FALSE; 2: posLogic, noLEDtest);
END Digital_Outputs
```

Siemens Building Technologies AG
Alte Landstrasse 411
CH-8708 Männedorf
Tel. +41 1 - 922 61 11
Fax +41 1 - 922 64 50
www.cerberus.ch