

1 TEST AND CONSTANT-MODULE	1-3
1.1 TEST-VAR BINARY (Test-Variable Binary)	1-4
1.2 TEST-VAR WORT (Test-Variable Wort)	1-4
1.3 TEST-VAR HEX (Test-Variable HexaDecimal)	1-4
1.4 TEST-VAR STRING (Test-Variable String)	1-4
1.5 TEST-VAR TIME (Test-Variable TIME)	1-4
1.6 CONSTANT BINARY	1-4
1.7 CONSTANT INTEGER	1-5
1.8 CONSTANT HEX	1-5
1.9 CONSTANT STRING	1-5
1.10 CONSTANT TIME	1-5
2 BINARY MODULE	2-7
2.1 CONSTANT	2-7
2.2 INVERTER	2-7
2.3 AND	2-8
2.4 OR	2-8
2.5 EXOR	2-8
2.6 RS FLIP-FLOP	2-8
2.7 JK FLIP-FLOP	2-9
2.8 EDGE RECOGNITION	2-11
3 TIME MODULE	3-12
3.1 TACT	3-12
3.2 MONO-FLOP	3-13
3.3 EDGE DELAY	3-14
4.2 FORWARD/BACKWARD COUNTER	4-18
5 ARITHMETIC MODULE	5-19
5.1 ADDITION	5-19
5.2 SUBTRACTION	5-19

5.3 MULTIPLICATION	5-19
5.4 DIVISION	5-20
5.5 COMPARISION	5-20

1 Test and Constant-Module

BINARY MODUL LEVEL:

T + K	BINARY
END	
TEST-VAR	BINARY
TEST-VAR	STRING
TEST-VAR	TIME
CONSTANT	BINARY
CONSTANT	STRING
CONSTANT	TIME
ALL	DEF
SET	PRIORITY
MODUL	DELETE
LINE	EDIT

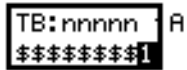
INT MODUL LEVEL:

T + K	INT
END	
INT	
TEST-VAR	WORT
TEST-VAR	HEX
TEST-VAR	STRING
TEST-VAR	TIME
CONSTANT	WORT
CONSTANT	HEX
CONSTANT	STRING
CONSTANT	TIME
DEF	ALL
SET	PRIORITY
MODUL	DELETE
LINE	EDIT

1.1 TEST-VAR BINARY

This modul is used for a binary source which can be changed via the programming tool or the ONLINE-Test.

Symbol:



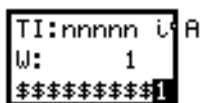
Functions logic:

nnnnnalphanumeric Name
\$\$\$\$\$\$\$Comment 8 char.acters

1.2 TEST-VAR INTEGER

This modul is used for a INTEGER-Data source which can be changed via the programming tool or the ONLINE-Test.

Symbol:



Functional logic:

nnnnnalphanumeric Name
\$\$\$\$\$\$\$Comment 9 Characters

1.3 TEST-VAR HEX

This modul is used for a HEXADECIMAL source which can be changed via the programming tool or the ONLINE-Test.

Symbol:



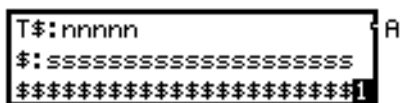
Functional logic:

nnnnnalphanumeric Name
W:HexaDecimal-Value
\$\$\$\$\$\$\$Comment 9 Characters

1.4 TEST-VAR STRING

This modul is used for a STRING source which can be changed via the programming tool or the ONLINE-Test.

Symbol:



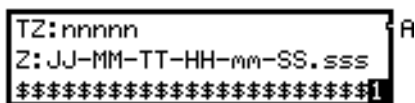
Functional logic:

nnnnnalphanumeric Name
ssssssssString-Daten (z.B.: abcDEF)
\$\$\$\$\$\$\$Comment 22 Characters

1.5 TEST-VAR TIME

This modul is used for a TIME-Data source which can be changed via the programming tool or the ONLINE-Test.

Symbol:



Functional logic:

nnnnnalphanumeric Name
Z:TIME-Daten
(z.B.: 87-01-27-08-15-36.238)
\$\$\$\$\$\$\$Comment 23 Characters

1.6 CONSTANT BINARY

This modul is used for a symbolic BINARY source.

Symbol:

```
KB:nnnnn A
$$$$$$$1
```

Functional logic:

nnnnnalphanumeric Name
\$\$\$\$\$\$\$Comment 8 Characters

1.7 CONSTANT INTEGER

This modul is used for a symbolic INT-Data source.

Symbol:

```
KI:nnnnn W A
W: 1
$$$$$$$1
```

Functional logic:

nnnnnalphanumeric Name
\$\$\$\$\$\$\$Comment 9 Characters

1.8 CONSTANT HEX

This modul is used for a symbolic HEXADECIMAL-Data source.

Symbol:

```
KH:nnnnn W A
W:0000
$$$$$$$1
```

Functional logic:

nnnnnalphanumeric Name
W:HexaDecimal-Value (z.B.: FF
oder 255 (entspricht 0xff))
\$\$\$\$\$\$\$Comment 9 Characters

1.9 CONSTANT STRING

This modul is used for a symbolic STRING-Data source.

Symbol:

```
K$:nnnnn A
$:ssssssssssssssssssss
$$$$$$$$$$$$$$$$$$$$1
```

Functional logic:

nnnnnalphanumeric Name
ssssssssString-Daten (z. B.: abcDEF)
\$\$\$\$\$\$\$Comment 22 Characters

1.10 CONSTANT TIME

This modul is used for a symbolic TIME-Data source.

Symbol:

```
KZ:nnnnn A
Z:JJ-MM-TT-HH-mm-SS.sss
$$$$$$$$$$$$$$$$$$$$1
```

Functional logic:

nnnnnalphanumeric Name
Z:TIME-Daten
(z.B.: 87-01-27-08-15-36.238)
\$\$\$\$\$\$\$Comment 23 Characters

2 BINARY MODULE

BINARY MODUL LEVEL:

BINARY	MODUL
END	
CONSTANT	
INVERTER	
AND	
OR	
EXOR	
RS FLIP-FLOP	
JK FLIP-FLOP	
EDGE	RECOGNITION
DEF	ALL
SET	PRIORITY
MODUL	DELETE
LINE	EDIT

2.1 CONSTANT

The CONSTANT Element has 2 Outputs, with „A1“ is always **logical 0** and „A2“ is always **logical 1**.

Symbol:



Logical table:

CONSTANT	
A1	A2
0	1

2.2 INVERTER

This modul inverts the binary incoming signal.

Symbol:



Logical table:

INVERTER	
E	A
0	1
1	0

2.3 AND

The output is logical „1“ when all the inputs are also logical „1“.

Symbol:



..... Comment 7 - 8 Characters

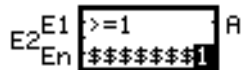
Logical table:

UND			
E1	E2	En	A
0	X	X	0
X	0	X	0
X	X	0	0
1	1	1	1

2.4 OR

Standard OR-Logic-module.

Symbol:



..... Comment 7 - 8 Characters

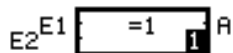
Logical table:

ODER			
E1	E2	En	A
1	X	X	1
X	1	X	1
X	X	1	1
0	0	0	0

2.5 EXOR

Standard EXOR-Logic-module.

Symbol:



Logical table:

EXODER		
E1	E2	A
0	1	1
1	0	1
1	1	0
0	0	0

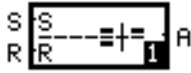
2.6 RS FLIP-FLOP

RS FLIP-FLOP were used to store a logic state. The priority can be setted with the modul SWITCH DEFs. The input „S“ is the Set-input. The input „R“ is the Reset-input.

RS FLIP-FLOP (Set-Priority):

The state at the output A is depending the the configuration at the input „S“ and „R“. For the output state see the mentioned table.

Symbol:



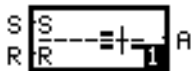
Logical table:

RS FLIP-FLOP SET		
S	R	A
1	X	1
0	1	0
0	0	A

RS FLIP-FLOP (Reset-Priority):

The state at the output A is depending the the configuration at the input „S“ and „R“. For the output state see the mentioned table.

Symbol:



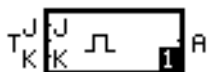
Logical table:

RS FLIP-FLOP RESET		
S	R	A
1	0	1
X	1	0
0	0	A

2.7 JK FLIP-FLOP

The logical table is mentioned as follows.

Symbol:

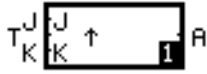


Logical table:

JK FLIP-FLOP PULS			
J	K	T	A
X	X	0	A
X	X	1	A
0	0	X	A
1	0	⌈	1
0	1	⌋	0
1	1	⌋	$\overline{A}$

JK FLIP-FLOP (triggert):

The logic table is mentioned as follows.



Symbol:

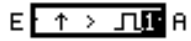
Logical table:

JK FLIP-FLOP FLANK			
J	K	T	A
X	X	0	A
X	X	1	A
0	0	X	A
1	0	↑	1
0	1	↑	0
1	1	↑	$\overline{A}$

2.8 EDGE RECOGNITION

If the input detects a rising edge then the output is logical „1“ for one PLC-cycle.

Symbol:



Logical table:

Edge recognition	
E	A
X ↑	0 ┐┌

3 TIME MODULE

This function is used to realise different timer functions.

BINARY MODUL LEVEL:

<u>TIME</u>	<u>BINARY</u>
END	
TACT	
MONO-FLOP	
EDGE DELAY	
DEF	ALL
SET	PRIORITY
MODUL	DELETE
LINE	EDIT

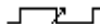
INTEGER MODUL LEVEL:

<u>TIME</u>	<u>INT</u>	
END		
<table border="1"><tr><td>INTEGER</td></tr></table>		INTEGER
INTEGER		
TACT		
MONO-FLOP		
EDGE DELAY		
DEF	ALL	
SET	PRIORITY	
MODUL	DELETE	
LINE	EDIT	

3.1 TACT

The tact-module generates a timer based tact at the modul output. The ON- and OFF-time is selectable via the „SWITCH DEF“ of the modul.

These switches are:

resp:	fix	
	variable	
TIMER-Basis:	Milliseconds	ms
	Seconds	sec

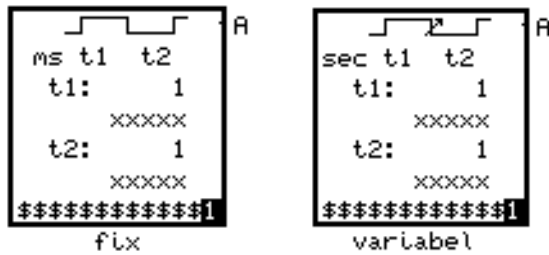
The switch „**TIME-Bases**“ is used to set the timer setvalue in **Milliseconds** and **Seconds**.

It is possible to define a variable state for each timer function. Thereby it is possible to adapt the setvalue via external control units.

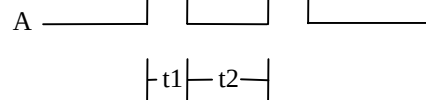
BINARY-Modul:

Value range t1 and t2: **INTEGER**

Symbol:



TIME diagramm:



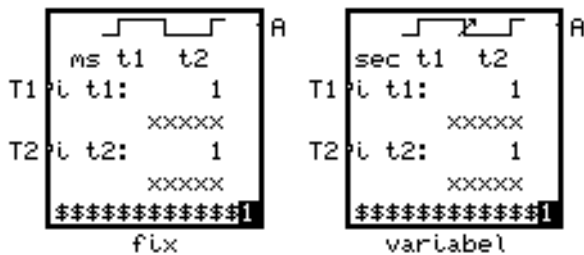
Here, „t1“ and „t2“ are internal values.

GANZZAHL-Modul (WORT):

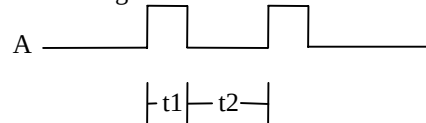
Value range for T1 und T2: **INTEGER**

Value range for t1 und t2: **INTEGER**

Symbol:



TIME diagramm:



The values „t1“ and „t2“ can be internal and external values. External values have a higher priority then internal values. To set external timer setvalues normally it is necessary to convert the value into the PLC-oriented timer format.

3.2 MONO-FLOP

The logical table is shown below:

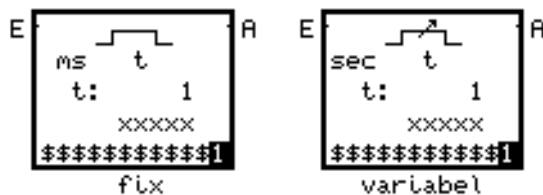
Modul switches are:

Typ:	fix	
	variable	
TIME-Basis:	Milliseconds	ms
	Seconds	sec

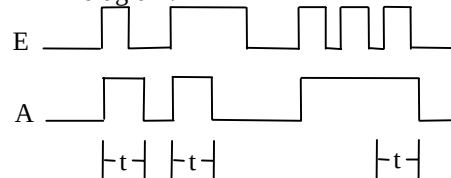
BINARY-Modul:

Value range for t: **INTEGER**

Symbol:



TIME diagramm:

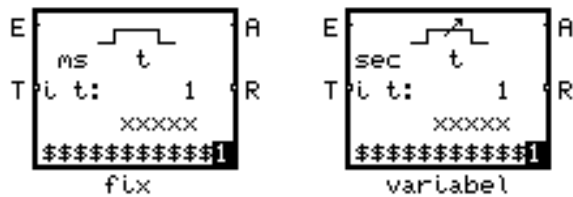


GANZZAHL-Modul (WORT):

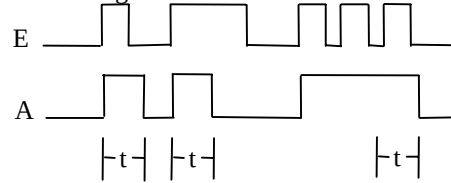
Value range for T: **INTEGER**

Value range for t: **INTEGER**

Symbol:



TIME diagramm:



3.3 EDGE DELAY

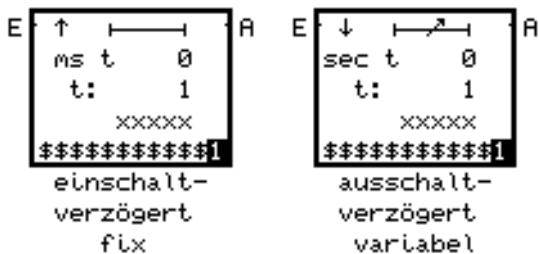
This modul gives his input to the output after the setted time. The set value can be defined with external or internal set values.

The switches are:

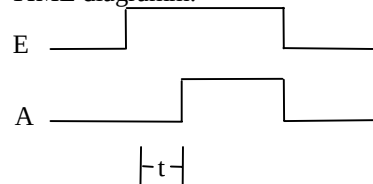
Typ:	fix	
	variable	
Verzögerungsart:	On delay	↑
	Off delay	↓
TIME base:	Milliseconds	ms
	Seconds	sec

BINARY-Modul:Value range for t: **INTEGER**

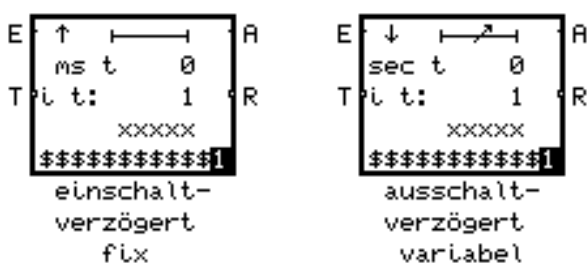
Symbol:



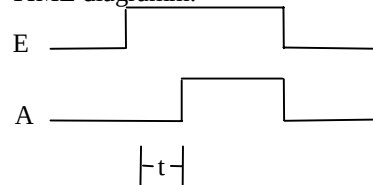
TIME diagramm:

**INTEGER-Modul:**Value range for T: **INTEGER**Value range for t: **INTEGER**

Symbol:



TIME diagramm:



Functional logic:

t: TIME (in sec oder ms)
 ↑ (↓) Delay mode
 \$\$\$\$ Comment 11 Characters

4 COUNTER MODUL

BINARY MODUL LEVEL:

<u>COUNTER</u>	<u>BINARY</u>
END	
BACKW. COUNTER	
DEF	ALL
SET	PRIORITY
MODUL	DELETE
LINE	EDIT

INT MODUL LEVEL:

COUNTER	INT	
END		
<table><tr><td>INT</td></tr></table>		INT
INT		
BACK		
FORW/BACKW		
DEF	ALL	
SET	PRIORITY	
MODUL	DELETE	
LINE	EDIT	

4.1 BACKWARD COUNTER

This modul is a backward counter. The set value is setted when the input „S“ gets logical „1“.

The switches are:

Typ:	fix	—
	variable	↗
Counter Mode:	Edge controled	↑
	Puls controled	⌋

Edge controled:

each puls at the input „D“ decrements the counter.

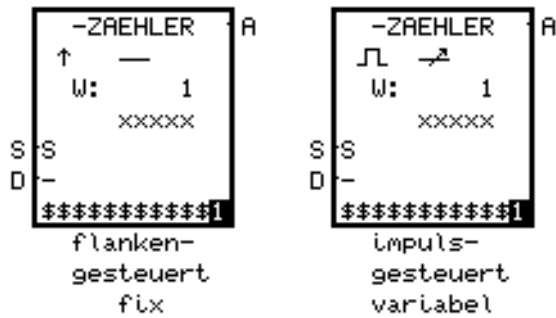
Puls controled:

With each PLC-cycle the counter will be decremented.

Reaches the present value 0, then the output „A“ get logical „1“.

BINARY-Modul: Value range for W: **INTEGER**

Symbol:



Functional logic:

xxxxx.....actuell Value (only in Simulation)
 S.....Set input
 DDecrement-Input
 #####Comment 11 Characters

GANZZAHL-Modul (WORT):Value range for W: **INTEGER**

Symbol:



Functional logic:

W:Zähl-Wert
 xxxxx.....aktueller Wert (nur in Simulation)
 S.....Setzeingang
 I.....Dekrement-Eingang
 #####Comment 11 Characters

4.2 FORWARD/BACKWARD COUNTER

This modul can be used for both function. Backward- and Upward counting.

INT-Modul (WORD):Value range for W und A: **INTEGER**

Symbol:



Functional logic:

W:Counter set value
 xxxxx.....actuel Value (only in Simulation)
 S.....Set input
 I.....Increment input
 DDecrement input
 R.....Reset input
 #####Comment 11 Characters

5 ARITHMETIC MODULE

INT MODUL LEVEL:

ARITHM.	INT
END	
WORT	
ADDITION	
SUBTRACTION	
MULTIPLICATION	
DIVISION	
COMPARISON	
ABS	
SGN	
MOD	
DEF	ALL
SET	PRIORITY
MODUL	DELETE
LINE	EDIT

5.1 ADDITION

Standard Addition in INT Level with Overflow detection at output OV.

INT-Modul (WORT):

Value range for E1 bis En sowie A: **INTEGER**

Symbol:

Functional logic:

$$\begin{matrix} E1 \\ E2 \\ \vdots \\ En \end{matrix} \begin{matrix} i1 \\ + \\ i2 \\ \vdots \\ i3 \end{matrix} \begin{matrix} A \\ OV \end{matrix}$$

$A = E1 + E2 + En$

\$\$\$\$\$\$Comment 5 – 6 Characters

5.2 SUBTRACTION

Standard Subtraction module in INT Level with Overflow detection.

INT-Modul (WORT):

Value range for E1, E2 und A: **INTEGER**

Symbol:

Functional logic:

$$\begin{matrix} E1 \\ E2 \end{matrix} \begin{matrix} i1 \\ - \\ i2 \end{matrix} \begin{matrix} A \\ OV \end{matrix}$$

$A = E1 - E2$

\$\$\$\$\$\$Comment 6 Characters

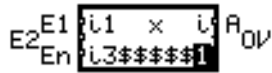
5.3 MULTIPLICATION

Standard Multiplcatin in INT Level with Overflow detection.

INT-Modul (WORD):

Value range for E1 bis En sowie A: **INTEGER**

Symbol:



Functional logic:

$$A = E1 * E2 * En$$

.....Comment 5 – 6 Characters

5.4 DIVISION

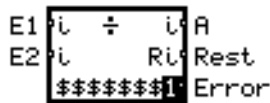
Standard Division in INT Level with Overflow and Division by Zero detection.

INT-Modul (WORT):

Value range for E1, E2, A und Rest: **INTEGER**

Symbol:

Functional logic:



$$A = \frac{E1}{E2}$$

.....Comment 7 Characters

Functional logic:

$$A = -E$$

5.5 COMPARISON

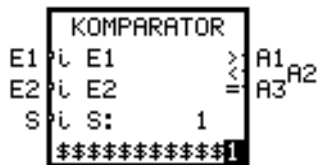
This module compares two input variables. The result is written to the binary outputs. The input „S“ define a range for the comparison.

INTEGER-Modul (WORT):

Value range for E1, E2 and S: **INTEGER**

Symbol:

Functional logic:



$A1 = 1$ if $E1 > (E2 + S)$
 $A2 = 1$ if $E1 < (E2 - S)$
 $A3 = 1$ if $|E1 - E2| \notin S$
 $S < 0$ is $S = 0$
 SRange
 #####Comment 11 Characters

A.