Concept

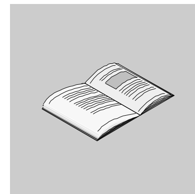
IEC block library

Part: DIAGNO

840 USE 494 00 eng Version 2.5

33001608.00

Table of Contents



About the book

At a Glance

Document Scope

This documentation is designed to help with the configuration of functions and function blocks.

Validity Note

This documentation applies to Concept 2.5 under Microsoft Windows 98, Microsoft Windows 2000 and Microsoft Windows NT 4.x.

Note: There is additional up to date tips in the README data file in Concept.

Related Documents

Title of Documentation	Reference Number
Concept Installation Instructions	840 USE 492 00
Concept User Manual	840 USE 493 00
Concept EFB User Manual	840 USE 495 00
Concept LL984 Block Library	840 USE 496 00

User Comments

We welcome your comments about this document. You can reach us by e-mail at TECHCOMM@modicon.com

General information about the DIAGNO function block library

I

Overview

At a Glance

This section contains general information about the DIAGNO function block library.

What's in this part?

This Part contains the following Chapters:

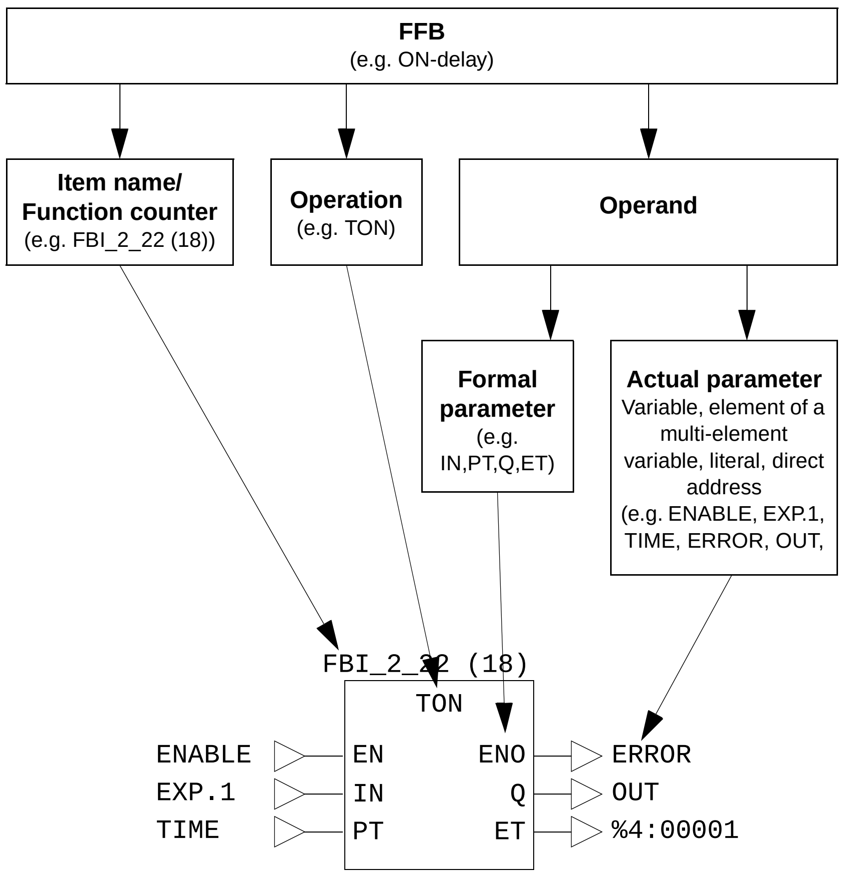
Parametering functions and function blocks

1

Parametering functions and function blocks

General

Each FFB consists of an operation, the operands needed for the operation and an instance name or function counter.



Operation

The operation determines which function is to be executed with the FFB, e.g. shift register, conversion operations.

Operand

The operand specifies what the operation is to be executed with. With FFBs, this consists of formal and actual parameters.

Formal/actual parameters

The formal parameter holds the place for an operand. During parametering, an actual parameter is assigned to the formal parameter.

The actual parameter can be a variable, a multi-element variable, an element of a multi-element variable, a literal or a direct address.

Conditional/ unconditional calls

"Unconditional" or "conditional" calls are possible with each FFB. The condition is realized by pre-linking the input EN.

- Expanded EN
  conditional calls (the FFB is only processed if EN = 1)
- EN collapsed
  unconditional calls (FFB is always processed)

> **Note:** If the EN input is not parametered, it must be collapsed. As non-parametered inputs are automatically used with a "0", the FFB would otherwise never be processed.

Calling functions and function blocks in IL and ST

Information on calling functions and function blocks in IL (Instruction List) and ST (Structured Text) can be found in the relevant chapters of the user manual.

Diagnostics

2

Overview

At a Glance

Two subjects are summarized under the subject diagnostics:
- System Diagnostics
- Process Diagnostics

What do we mean by system diagnostics?

System diagnostics deals with the analysis of the PLC status. It is part of the delivered system and always works without any programming.

What do we mean by process diagnostics?

The process diagnostics observes the external PLC environment and recognizes whether or not the process devices are functioning in the specified mode.

What's in this Chapter?

This Chapter contains the following Maps:

System diagnostics

System diagnostics capabilities

Concept offers the following selftest capabilities:
- Gathering of error conditions from bus modules
- I/O
- Communication
- Comparing of programmed configuration with current configuration of bus modules
- Parameter check of the function blocks

Mode of operation for system diagnostics

These system diagnoses are part of the delivered system and always work without any programming. Error situations that arise outside the system, specifically in Elementary function blocks, are automatically saved and will be displayed upon request. A message will automatically appear if there is at least one error message. Error messages have a number that identifies their type.

Message display

Error messages from the programming unit are displayed as text in the "Event viewer". Double-clicks on the list text open the image of the section, where the respective EFB is located. Status information is also displayed automatically. Exceeding the time limits in the Steps of the SFC utilizes the same options, except that the step name is displayed instead of the EFB item name and that the open section is the respective SFC section.

Process diagnostics

Process diagnostics capabilities

The process diagnostics observe the external PLC environment and report whether or not the process devices are functioning in default mode. The default behavior is set in the programming phase of the system and is reflected in the diagnostic functions of the runtime system. These diagnostic functions operate with a small subset of the input/output signals of the total process. In the actual process, this subset represents physically existing devices such as cylinders or motors together with their assigned limit switches.

Mode of operation for process diagnostics

The process diagnostics is implemented with the use of EFBs. One special EFB is provided for each diagnostics type. The USR (user runtime system) downloads each EFB with the current parameters, which could also be the result of a link, only once. They can be executed several times and have separate data areas for each Instance.

Diagnostic base EFBs

The diagnostic base EFBs are in the group "Diagnostics".
The following diagnostic base EFBs are available:

- ACT_DIA (See *ACT_DIA: Action diagnostics, p. 19*)
  Action diagnostics with optional motor-like behavior or pulse behavior
- DYN_DIA (See *DYN_DIA: Dynamic diagnostics, p. 27*)
  Dynamic diagnostics
- GRP_DIA (See *GRP_DIA: Signal group monitoring, p. 41*)
  Signal group monitoring
- LOCK_DIA (See *LOCK_DIA: Locking diagnostics, p. 45*)
  Locking diagnostics without reaction input
- PRE_DIA (See *PRE_DIA: Monitoring of process requirements, p. 51*)
  Monitoring of process requirements
- REA_DIA (See *REA_DIA: Reaction diagnostics, p. 55*)
  Reaction diagnostics

Extended diagnostics EFBs

The extended diagnostics EFBs are in the group "Extended".
These function blocks can be used for visualization of the diagnostics in conjunction with one of the following programs:
- Diagnostics Viewer in Concept (**Online** → **Online-Diagnostics...**)
- various diagnostics software

> **Note:** This additional diagnostic information can only be utilized when using function blocks in the FBD (Function Block Dialog) programming language.

The following extended diagnostics EFBs are available:
- XACT (See *XACT: Extended locking/action diagnostics, p. 59*)
  Extended combination of locking and action diagnostics
- XACT_DIA (See *XACT_DIA: Extended action diagnostics, p. 71*)
  Extended action diagnostics with optional motor-like behavior or pulse behavior
- XDYN_DIA (See *XDYN_DIA: Extended dynamic diagnostics, p. 79*)
  Extended dynamic diagnostics
- XGRP_DIA (See *XGRP_DIA: Extended signal group monitoring, p. 85*)
  Extended signal group monitoring
- XLOCK (See *XLOCK: Extended locking diagnostics, p. 65*)
  Extended locking diagnostics with reaction input
- XLOCK_DIA (See *XLOCK_DIA: Extended locking diagnostics, p. 89*)
  Extended locking diagnostics without reaction input
- XPRE_DIA (See *XPRE_DIA: Extended process requirement monitoring, p. 95*)
  Extended monitoring of process requirements
- XREA_DIA (See *XREA_DIA: Extended reaction diagnostics, p. 99*)
  Extended reaction diagnostics

EFB descriptions

II

Overview

At a Glance

These EFB descriptions are documented in alphabetical order.

> **Note:** The number of certain EFB inputs can be increased up to a maximum of 32 by vertically modifying the size of the FFB symbol. Refer to the description of the individual EFBs to determine which ones are concerned.

What's in this part?

This Part contains the following Chapters:

ACT_DIA: Action diagnostics

3

Overview

At a Glance

This chapter describes the ACT_DIA block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The function block ACT_DIA is used for action diagnostics.
Action diagnostics is initiated when the defined action becomes active. This action initiates an operation in the process. This operation has to trigger a set reaction. This reaction mostly occurs with a set delay. However, if the reaction does not occur within the tolerance time DTIME, an error situation arises and the error output ERR becomes active. Contrary to the Locking diagnostics, in which the trigger of the diagnostics must remain active at all times, the behavior of the trigger (action) in action diagnostics can vary.
There are 3 different types of behavior:

- M behavior
- I behavior
- MI behavior

These alternatives vary in the behavior of the diagnostics if the action signal becomes "0" before an authorized value is placed at the reaction input.
The monitoring is performed cyclically. The activation of the diagnostics and at the same time the distribution of the cycle load can be achieved through the enable signal "ED".

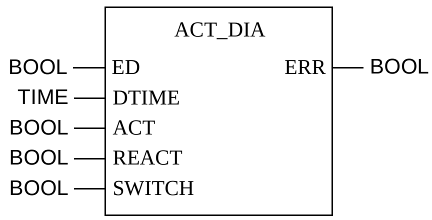
Note: NEVER use diagnostic EFBs in DFBs.

EN and ENO can be projected as additional parameters.

Representation

Symbol

Block representation:



Parameter description

Block parameter description:

Parameter	Data type	Meaning
ED	BOOL	Enable diagnostics
DTIME	TIME	Tolerance time
ACT	BOOL	Action signal
REACT	BOOL	Reaction signal
SWITCH	BOOL	M/I switch; 0: M behavior, 1: I behavior, 0/1: MI behavior
ERR	BOOL	Error message; 0: no error; 1: Error

Detailed description

Parametering

In order to control the different modes of behavior (M, I, MI), the appropriate value must be set at SWITCH.

Behavior	SWITCH
M behavior	0
I behavior	1
MI behavior	0 -> 1 (value modification within selected time)

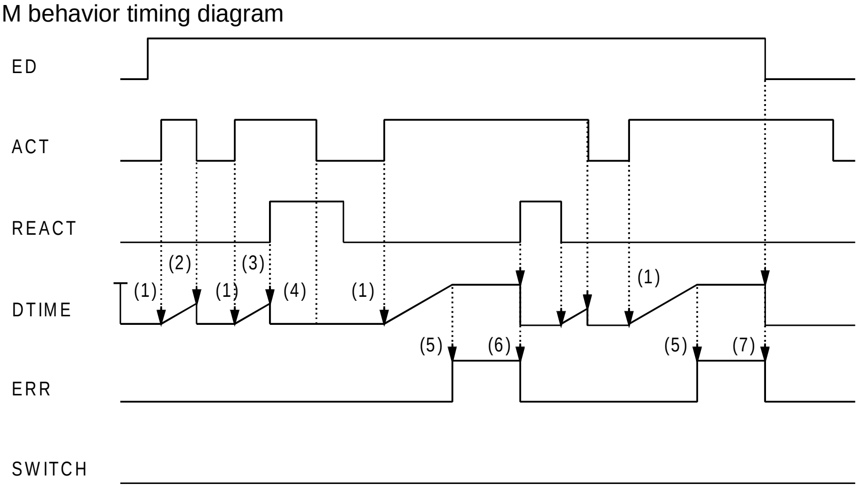
ACT_DIA: M behavior

Motor-like behavior

If the ACT input becomes "1" and REACT does not, the internal counter will be started.
If the action becomes inactive during processing, the monitoring time is stopped/ reset or in the event of an error, the error processing is stopped.
When the default time at the DTIME input has expired, the ERR output will display an error; it remains active until ACTION becomes "0", REACT becomes "1" or the diagnostics is deactivated.
If the tolerance time (DTIME) is entered as "0", an error message is displayed as soon as an error situation occurs.
An example for the process of an action diagnostics with M behavior is given in the timing diagram.

Timing diagram

M behavior timing diagram



1. The internal time will start when ACT is "1" and REACT is "0".
2. The internal time is stopped/reset when ACT is "0".
3. The internal time is stopped/reset when REACT becomes "1".
4. Once the reaction has been detected, it is insignificant whether or not ACT is active.
5. If the internal time reaches the DTIME value, an error is reported.
6. The error is cancelled and the internal time stopped/reset when REACT becomes "1".
7. The error is cancelled and the internal time stopped/reset, if the enable signal ED is "0".

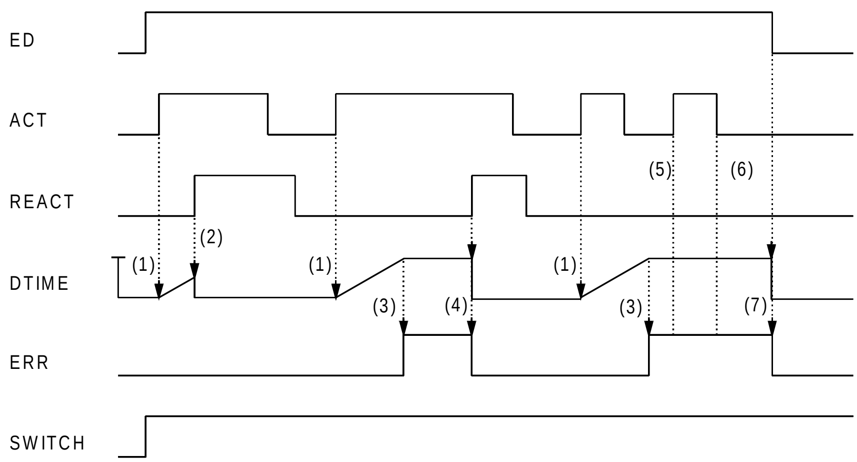
ACT_DIA: I behavior

Pulse behavior

After a transition of the action signal has been detected, diagnostics is activated and the monitoring time will start. The valency of the action signal is no longer significant. When the default time at the DTIME input has expired, the ERR output will display an error; it remains active until REACT becomes "1" or diagnostics is deactivated. The diagnostics will only be terminated (different from the M behavior) with the incoming defined reaction. In order to allow the diagnostics to be terminated in case of error, the ED enable signal has to be projected.
If the tolerance time (DTIME) is entered as "0", an error message is displayed as soon as an error situation occurs.
An example for the process of an action diagnostics with I behavior is given in the timing diagram.

Timing diagram

I behavior timing diagram



1. The internal time will start when ACT is "1" and REACT is "0".
2. The internal time is stopped/reset when REACT becomes "1".
3. If the internal time reaches the DTIME value, an error is reported.
4. The error is cancelled and the internal time stopped/reset when REACT becomes "1".
5. If the action diagnostics are still in progress (e.g. error handling), a positive transition of the action has no significance.
6. If the action diagnostics are still in progress (e.g. error handling), a negative transition of the action has no significance.
7. The error is cancelled and the internal time stopped/reset, if the enable signal ED is "0".

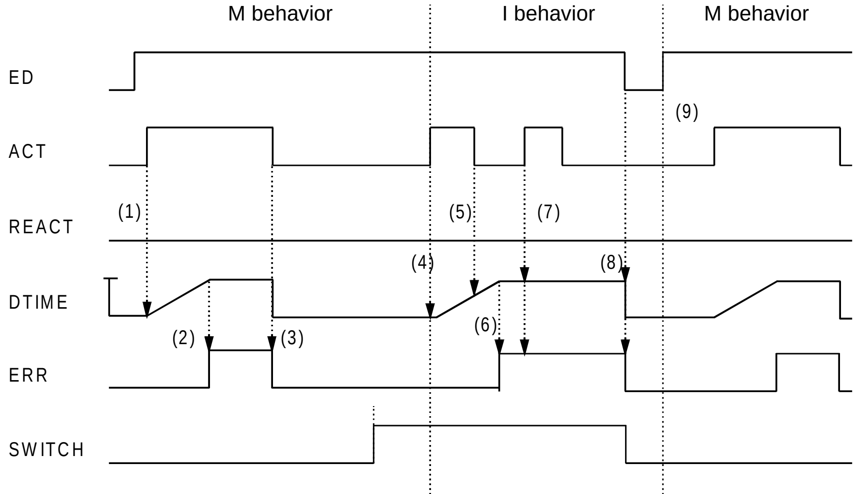
ACT_DIA: MI behavior

MI behavior

In the MI behavior, the monitoring begins with M behavior. If the SWITCH signal becomes active during monitoring (transition), the diagnostics switches to I behavior. This is a one-time switch and it is not possible to change back to M behavior during this monitoring cycle.
Since action diagnostics with I behavior can only be terminated via the defined reaction or the ED enable signal, the enable signal has to be projected in the MI behavior as well.
If the tolerance time (DTIME) is entered as "0", an error message is displayed as soon as an error situation occurs.
An example for the process of an action diagnostics with MI behavior is given in the timing diagram.

Timing diagram

MI behavior timing diagram



1. The internal time will start when ACT is "1" and REACT is "0".
2. If the internal time reaches the DTIME value, an error is reported.
3. With M behavior, the error will be cancelled, and the internal time stopped/reset when ACT becomes "0".
4. If SWITCH is "1" and ACT becomes "1", the diagnostics will switch from M behavior to I behavior. The internal time will also start when ACT is "1" and REACT is "0".
5. If the action diagnostics is still in progress (e.g. internal time started) during I behavior, a negative transition of the action has no significance.
6. If the internal time reaches the DTIME value, an error is reported.
7. If the action diagnostics is still in progress (e.g. internal time started) during I behavior, a positive transition of the action has no significance.
8. The error is cancelled and the internal time stopped/reset, if the enable signal ED is "0".
9. If the enable signal ED returns to "1" or REACT becomes "1", a switch from I behavior to M behavior occurs.

DYN_DIA: Dynamic diagnostics

4

Overview

At a Glance

This chapter describes the DYN_DIA block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The function block DYN_DIA is used for dynamic diagnostics.
Some processes require that LOCK_DIA (Locking diagnostics), ACT_DIA (Action diagnostics) and REA_DIA (reaction diagnostics) are combined into one unit that monitors the momentary condition of the diagnostics. This is only possible using a special function block, which internally manages the current diagnostics status.
To prevent this function block becoming too complex, only one ED enable signal and one ERR error output were defined.
The monitoring is performed cyclically. Activation of the diagnostics and, at the same time, distribution of the cycle load can be achieved through the enable signal ED.

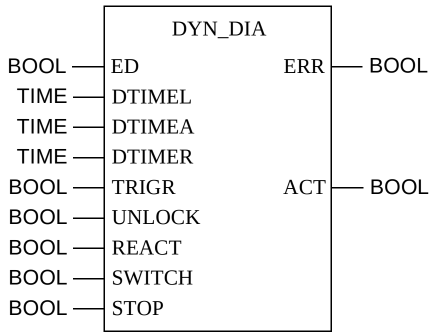
Note: NEVER use diagnostic EFBs in DFBs.

EN and ENO can be projected as additional parameters.

Representation

Symbol

Block representation:



Parameter description

Block parameter description:

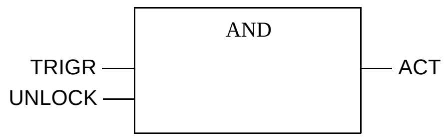
Parameter	Data type	Meaning
ED	BOOL	Enable diagnostics
DTIMEL	TIME	Tolerance time LOCK_DIA (locking diagnostics)
DTIMEA	TIME	Tolerance time ACT_DIA (action diagnostics)
DTIMER	TIME	Tolerance time REA_DIA (reaction diagnostics)
TRIGR	BOOL	Trigger
UNLOCK	BOOL	Locking
REACT	BOOL	Reaction signal
SWITCH	BOOL	M/I switch; 0: M behavior, 1: I behavior, 0/1: MI behavior
STOP	BOOL	Stop signal
ERR	BOOL	Error message; 0: No error; 1: Error
ACT	BOOL	Action enabling

Detailed description

Parameetering

Note: The ACT output is created from TRIGR and UNLOCK with a logical AND. Other inputs (e.g. ED) have no effect on this.

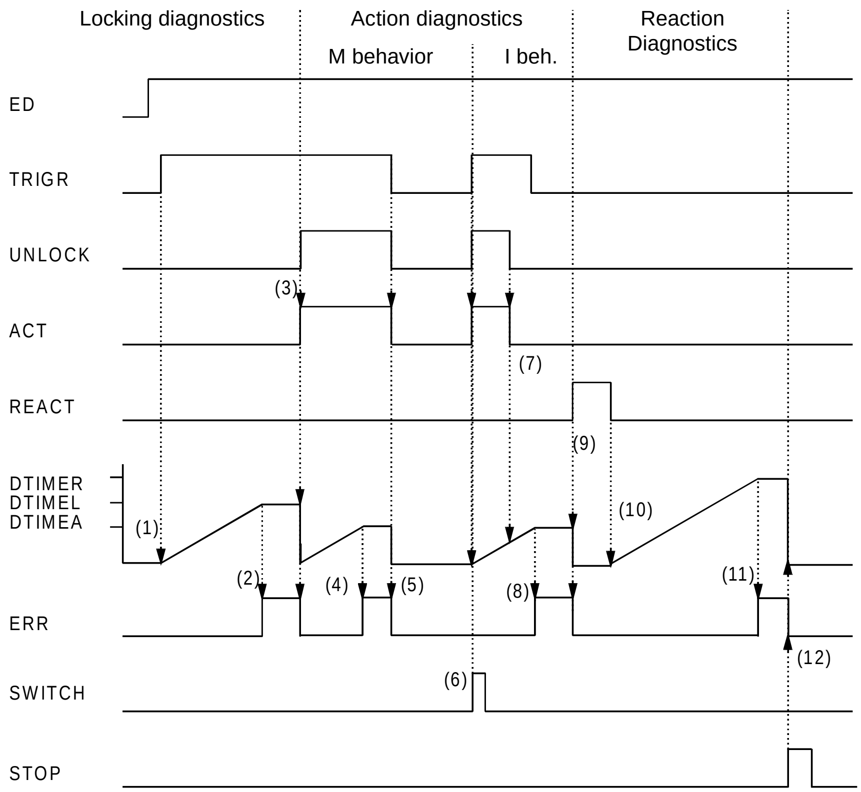
Representation of the relevant inputs for the ACT output



The parametering of the individual diagnostics types can be found in the descriptions for LOCK_DIA, ACT_DIA and REA_DIA.
An individual tolerance time (DTIMEL, DTIMEA, DTIMER) can be parametered for every diagnostics type.
An example for the process of dynamic diagnostics is given in the timing diagram.

Timing diagram

Timing diagram for dynamic diagnostics



1. The internal time starts when TRIGR is "1" and UNLOCK is "0".
2. If the internal time reaches the DTIMEL value, an error will be reported.
3. If UNLOCK becomes "1", the error will be cancelled, the internal time is stopped/ reset, and ACT becomes "1". Activating the action switches to the action diagnostics. As the reaction has still not occurred, the internal time is started.
4. If the internal time reaches the DTIMEA value, an error will be reported.
5. With M behavior, the error will be cancelled, and the internal time stopped/reset when ACT becomes "0".
6. If SWITCH becomes "1" and ACT is "1", a switch from M behavior to I behavior occurs. The internal time will also start when ACT is "1" and REACT is "0".
7. If the action diagnostics is still in progress (e.g. internal time started), a negative transition of the action has no significance.
8. If the internal time reaches the DTIMEA value, an error will be reported.
9. If REACT becomes "1", the internal time is stopped/reset. By activating the reaction, the reaction diagnostics is switched.
10. The internal time will start when REACT becomes "0".
11. If the internal time reaches the DTIMER value, an error will be reported.

12. The error will be cancelled and the internal time is stopped/reset when STOP becomes "1". By activating the stop signal, the locking diagnostics is switched again.

ERR2HMI: Error to HMI

5

Overview

At a Glance

This chapter describes the ERR2HMI block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The function block ERR2HMI is used to make error data from the internal PLC error buffer, which is detected by the diagnostic EFBs and the sequential function chart, available for diagnostic display in visualization.
The PLC error buffer is part of the runtime system and provides no interface for direct external access. EFBs, that read data from the error buffer and forward it to the corresponding receiver, are used to make the error data available for visualization.
The diagnostics communicates with the block via both the PCV_IN and PCV_OUT data structures. Jobs are assigned to the block in PCV_IN. The result is written to PCV_OUT by the block. PCV_OUT is read by the diagnostic display.
Only one ERR2HMI block may be projected per diagnostic display.

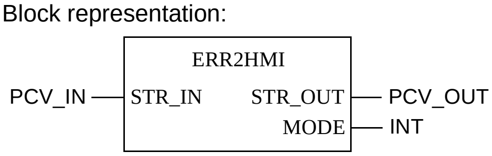
Note: NEVER use diagnostic EFBs in DFBs.

EN and ENO can be projected as additional parameters.

Representation

Symbol

Block representation:



Parameter description

Block parameter description:

Parameters	Data type	Meaning
STR_IN	PCV_IN	Input data structure
STR_OUT	PCV_OUT	Output data structure
MODE	INT	0 = active, 1 = communication interrupted

Description of the PCV_I block elements:

Element	Data type	Meaning
Auftrag	INT	Processing job for block
Parameter	INT	Parameters for specific jobs, e.g. error selection

Description of the PCV_OUT block elements:

Element	Data type	Meaning
response	INT	Processing status: 0 = o.K.
counter	INT	Write counter of error buffer
st_feld [1] ... st_feld [64]	INT	Status fields for the 64 possible error entries
laenge	INT	Length of error entry
klasse	INT	Error class
typ	INT	Error type
station	INT	Drop number
q_status	INT	Acknowledgment status
m_status	INT	Message status
t_kommt	DINT	Time stamp when error event occurs
t_kommt_ms	INT	Time stamp when error event occurs
blob_id	INT	Internal ID character
t_geht	DINT	Time stamp when error event ends
t_geht_ms	INT	Time stamp when error event ends
scan_index	INT	Reference to diagnostic block
blob_adr	DINT	internal address
blob_gen_time	DINT	Time of generation
trans_ID	INT	References to transition
anz_signale	INT	Number of error signals
fehler_liste [1] ... fehler_liste [20]	INT	Reference to faulty signals

Note: Both data structures, PCV_IN and PCV_OUT, are only used for communication with the diagnostic display and may **not be manipulated** by the user

ERRMSG: Message for error buffer overflow

6

Overview

At a Glance

This chapter describes the ERRMSG block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The function block ERRMSG is used to display error buffer status information.

A display shows whether there has been a buffer overflow and a counter measures the number of lost entries.

These values are standardized during the initialization of the error buffer (load program, reset error buffer).

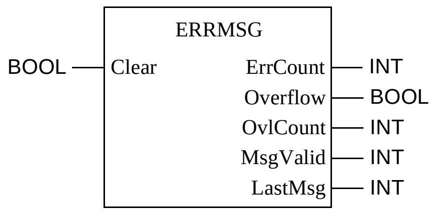
Note: NEVER use diagnostic EFBs in DFBs.

EN and ENO can be projected as additional parameters.

Representation

Symbol

Representation of the block:



Parameter description

Block parameter description:

Parameters	Data type	Meaning
Clear	BOOL	A 0 -> 1 edge standardizes the inputs to: ● ErrCount = current value ● Overflow = 0 ● OvlCount = 0 ● MsgValid = 0 ● LastMsg = 0
ErrCount	INT	Shows the number of current entries in the error buffer
Overflow	BOOL	1 = An error should be entered in the buffer, but there is no space available for this entry. The output is reset when an error entry is deleted, thus making room for a new entry.
OvlCount	INT	Shows how often the overflow output has been set, i.e. how many entries into the error buffer were lost.
MsgValid	INT	1 = The error displayed in LastMsg is current. 0 = The error displayed in Last Msg is no longer valid.
LastMsg	INT	Display of the last Status (See *Error buffer status, p. 40*) encountered in the error buffer. The display remains until a standardization (input clear) is performed. The display's validity is shown in MsgValid.

Detailed description

Function description

Error messages and error withdrawal can appear several times in one cycle and the queries to the error buffer entries are added. This leads to an update of the output couple LastMsg and MsgValid respectively. The EFB reads the displayed values from the buffer at the point of execution and thus displays a momentary record which cannot make every state in the buffer available. As the number of overflows (data loss) is summed up at the OvlCount output, this value can still be read later.

Error buffer status

The following messages have been defined:

Value	Meaning	Message in the event viewer
0	no error	-
-4601	Buffer full	Insufficient memory for error buffer
-4602	Diagnostics not installed	Error buffer is not present
-4603	Memory error	Memory management error
-4604	Error index invalid	Incorrect error ID
-4605	MMI not logged in	Incorrect MMI ID
-4606	MMI log-on not possible (too many MMI)	MMI calculation terminated
-4607	Diagnostics data (BLOB) not found	BLOB not loaded
-4608	Block instance not found	No error data present for this element

GRP_DIA: Signal group monitoring

7

Overview

At a Glance

This chapter describes the block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The GRP_DIA function block is used for signal group monitoring.
The monitoring is performed cyclically. The activation of the Diagnostics and thereby the distribution of the cycle load can be achieved through the enable signal "ED".

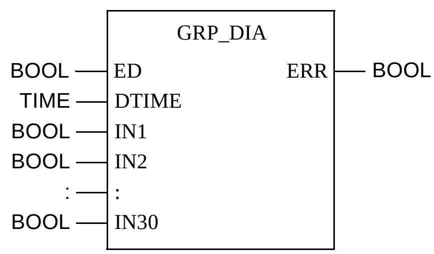
Note: NEVER use diagnostic EFBs in DFBs.

EN and ENO can be projected as additional parameters.

Representation

Symbol

Block representation:



Parameter description

Description of block parameter:

Parameters	Data type	Meaning
ED	BOOL	Enable diagnostics
DTIME	TIME	Tolerance time
IN1	BOOL	1. Signal
IN2	BOOL	2. Signal
:	:	:
IN30	BOOL	30. Signal
ERR	BOOL	Error message; 0: no error; 1:error

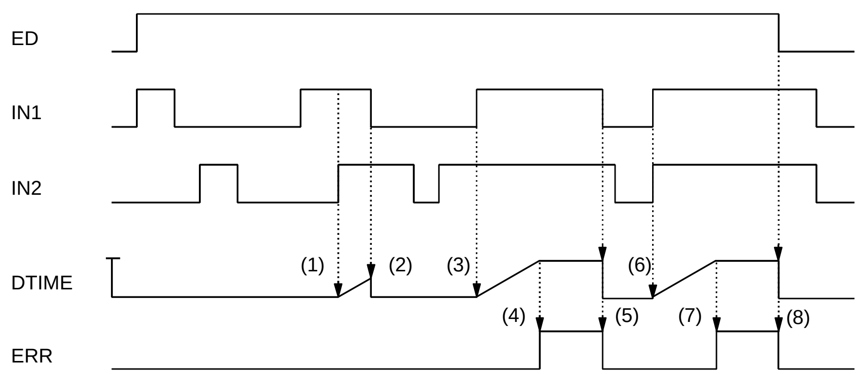
Detailed description

Parametering

The inputs IN1 and IN2 are monitored whether more than one input is "1". Deactivating the diagnostics or the attached correct values at the inputs will reset the internal counter to "0".
When the default time at the DTIME input has expired, the ERR output displays an error; it remains active until less than two inputs are "1" or the diagnostics is deactivated.
If a tolerance time (DTIME) of "0" is entered, an error message appears immediately if more than one input becomes "1".
An example for the process of signal group monitoring is given in the timing diagram

Timing diagram

Signal group monitoring timing diagram



1. The internal time is started when IN1 and IN2 simultaneously become"1".
2. The internal time is stopped/reset when IN1 becomes "0".
3. The internal time is started when IN1 and IN2 simultaneously become"1".
4. If the internal time reaches the DTIME value, an error will be reported.
5. The error is cancelled and the internal time is stopped/reset when IN1 becomes "0".
6. The internal time is started when IN1 and IN2 simultaneously become"1".
7. If the internal time reaches the DTIME value, an error will be reported.
8. The error is cancelled and the internal time stopped/reset, if the enable signal ED is "0".

LOCK_DIA: Locking diagnostics

8

Overview

At a Glance

This chapter describes the LOCK_DIA block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The LOCK_DIA function block is used for locking diagnostics and to enable the action.
Locking Diagnostics is activated when the input with the TRIGR signals becomes active.
In control networks the trigger signal TRIGR (e.g. step counter, manual key) does not necessarily initiate the execution of an action directly, but is generally combined with locks from the process. It is therefore possible that the action ACT only becomes active after a time delay or not at all. It is the task of lock diagnostics to check whether UNLOCK is enabled within a tolerance time DTIME when the trigger signal is active. In this case the lock diagnostics enables the action ACT. In this instance the trigger signal TRIGR must be active throughout the entire time. An error situation exists if the lock enabler UNLOCK does not appear within the time period, (lock not free). In this instance the action output ACT does not become active and the error output ERR is set. This error message is terminated when the trigger signal TRIGR is inactive or the lock enabler UNLOCK becomes active.
The lock diagnostics is terminated with an active action output ACT.
The monitoring is performed cyclically. Activation of the diagnostics and thereby distributing the cycle load can be achieved through the enable signal ED. The ED enable signal refers only to the activation of the diagnostics and has no effect on the ACT output.

Note: Contrary to the LOCK-DIA function block, the XLOCK function block has a REACT input that allows the ACT output to be switched off or prevents its activation, respectively, without a locking error being reported.

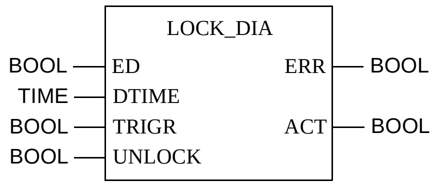
Note: NEVER use diagnostic EFBs in DFBs.

The parameters EN and ENO can additionally be projected.

Representation

Symbol

Block representation:



Parameter description

Block parameter description:

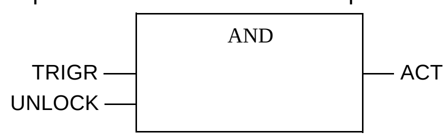
Parameters	Data type	Meaning
ED	BOOL	Enable diagnostics
DTIME	TIME	Tolerance time
TRIGR	BOOL	Trigger signal
UNLOCK	BOOL	Lock
ERR	BOOL	Error message; 0: no error; 1: Error
ACT	BOOL	Action output

Detailed description

Parameterening

Note: The output is created with a logical AND from TRIGR and UNLOCK. Other inputs (e.g. ED) have no effect on this.

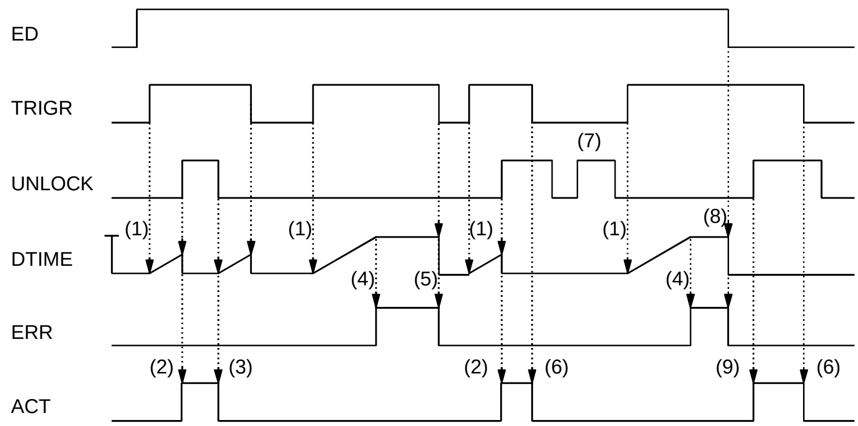
Representation of the relevant inputs for the ACT output



If the TRIGR input (trigger signal) becomes "1" and UNLOCK does not, the internal counter will be started.
When the default time at the DTIME input has expired, the ERR output will display an error; it remains active until TRIGR becomes "0", ACT becomes "1" or the diagnostic is deactivated.
If the trigger time (DTIME) is entered as "0", an error message is displayed as soon as an error situation occurs.
An example for the process of a lock diagnostic is given in the timing diagram.

Timing diagram

Locking diagnostics timing diagram



1. The internal time starts when TRIGR is "1" and UNLOCK is "0".
2. The internal time is stopped/reset and ACT becomes "1" when UNLOCK becomes "1".
3. ACT becomes "0" when UNLOCK becomes "0".
4. If the internal time reaches the DTIME value, an error will be reported.
5. The error is cancelled and the internal time stopped/reset when TRIGR becomes "0".
6. ACT becomes "0" when TRIGR becomes "0".
7. If UNLOCK is "1" and TRIGR is "0", the internal time does not start.
8. The error is cancelled and the internal time stopped/reset, if the enable signal ED becomes "0".
9. If TRIGR and UNLOCK are "1" and ED is "0", action becomes "1". ED has no effect on the ACT signal.

PRE_DIA: Monitoring of process requirements

9

Overview

At a Glance

This chapter describes the PRE_DIA block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The function block PRE_DIA is used for the Monitoring process requirements. Process requirements are process characteristics that are absolutely necessary for the operation of the machine or system (e.g. coolant, emergency stop). General requirements are for example requirements for machine operating modes or basic settings
The absence of such requirements is monitored. The monitoring is carried out cyclically. The activation of the diagnostics and thereby the distribution of the cycle load can be achieved through the enable signal ED.
The number of inputs IN can be increased up to 30 by vertically modifying the size of the block.

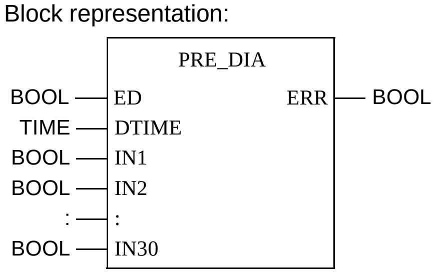
Note: NEVER use diagnostic EFBs in DFBs.

EN and ENO can be projected as additional parameters.

Representation

Symbol

Block representation:



Parameter description

Block parameter description:

Parameters	Data type	Meaning
ED	BOOL	Enable diagnostics
DTIME	TIME	Tolerance time
IN1	BOOL	1. Process requirement
:	:	:
IN30	BOOL	30. Process requirement
ERR	BOOL	Error message; 0: no error; 1: Error

Detailed description

Parametering

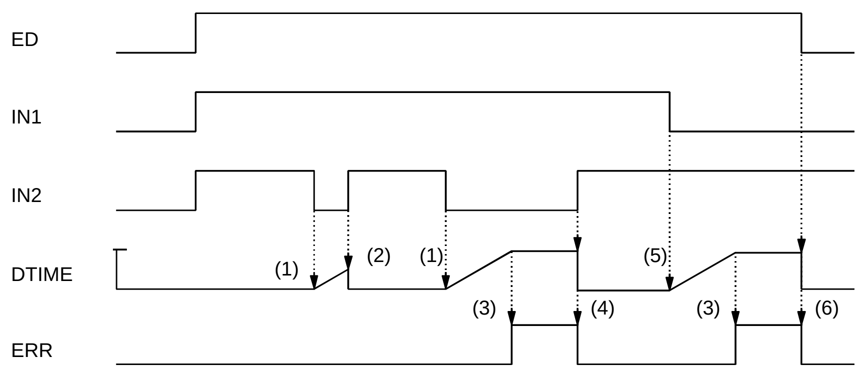
If at least one of the signals connected to INx becomes "0" and the diagnostics is active, the internal counter will be started.

> **Note:** Please note that all visible and unlinked inputs are automatically assigned a "0", i.e. create only as many IN inputs as are actually needed.

The deactivation of the diagnostics or of the attachment of the correct input value stops the counter (the requirements may contain errors during the tolerance time) and sets the counter back to "0".
When the default time at the DTIME input has expired, the ERR output displays an error; it remains active until the requirements are "1" or the diagnostic is deactivated.
If the tolerance time entered is DTIME "0", there is an immediate error message when the static conditional values (INx) become "0".
An example showing the process of monitoring the process requirements can be found in the timing diagram.

Timing diagram

Timing diagram monitoring process requirements



1. The internal time will start when IN2 becomes "0".
2. The internal time is stopped/reset when IN2 becomes "1".
3. If the internal time reaches the DTIME value, an error will be reported.
4. The error is cancelled and the internal time is stopped/reset when IN2 becomes "1".
5. The internal time will start when IN1 becomes "0".
6. The error is cancelled and the internal time stopped/reset, if the enable signal ED becomes "0".

REA_DIA: Reaction diagnostics

10

Overview

At a Glance

This chapter describes the REA_DIA block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The function block REA_DIA is used for reaction Diagnostics.
Once the expected reaction has occurred in the Action diagnostic, the reaction diagnostics check whether the process contains the status.
The process reaction, defined as a term or a signal, is checked through the reaction diagnostics to determine whether the status is stable. During technical processes, it is possible that reactions change momentarily (e.g. hitting limit switches). In order for the reaction diagnostics not to activate the error message ERR directly in such a case, a tolerance time DTIME can be defined. An error signal occurs if this time is exceeded. The error signal becomes inactive when the reaction returns to the setpoint status or when the stop condition is met.
The stop condition terminates reaction diagnostics.
Monitoring is performed cyclically. Activation of the diagnostics and, at the same time, distribution of the cycle load can be achieved through the enable signal ED.

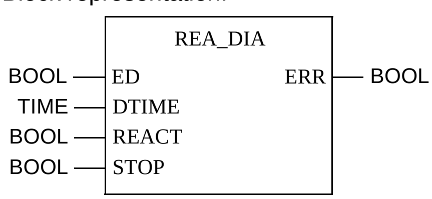
Note: NEVER use diagnostic EFBs in DFBs.

The parameters EN and ENO can additionally be projected.

Representation

Symbol

Block representation:



Parameter description

Block parameter description:

Parameters	Data type	Meaning
ED	BOOL	Enable diagnostics
DTIME	TIME	Tolerance time
TRIG	BOOL	Trigger
REACT	BOOL	Reaction signal
STOP	BOOL	Stop signal
ERR	BOOL	Error message; 0: no error; 1: Error

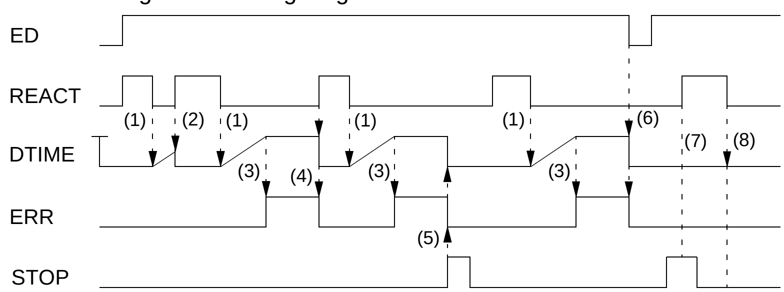
Detailed description

Parametering

If the REACT input becomes "0", the internal counter will be started.
When the default time at the DTIME input has expired, the ERR output will display an error; it remains active until REACT becomes "1", STOP becomes "1" or the diagnostic is deactivated.
If the tolerance time (DTIME) is entered as "0", an error message is displayed as soon as an error situation occurs.
An example for the process of a reaction diagnostic is given in the timing diagram.

Timing diagram

Reaction diagnostics timing diagram



1. The internal time will start when REACT becomes "0".
2. The internal time is stopped/reset when REACT becomes "1".
3. If the internal time reaches the DTIME value, an error will be reported.
4. The error is cancelled and the internal time stopped/reset when REACT becomes "1".
5. The error will be cancelled and the internal time is stopped/reset when STOP becomes "1".
6. The error is cancelled and the internal time stopped/reset, if the enable signal ED becomes "0".
7. If REACT becomes "1", when STOP is "1", the reaction diagnostic is not started.
8. If subsequently REACT becomes "0", the internal time is not started, even if STOP is "0" again.

XACT: Extended locking/action diagnostics

11

Overview

At a Glance

This chapter describes the XACT block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The XACT function block provides a combination of locking and action Diagnostics. The **Locking diagnostics** are activated when the input with the TRIGR signals becomes active.
In control networks the trigger signal TRIGR (e.g. step counter, manual key) does not necessarily initiate the execution of an action directly, but is generally combined with locks from the process. It is therefore possible that the action ACT only becomes active after a time delay or not at all. It is the task of lock diagnostics to check whether UNLOCK is enabled within a tolerance time (DTIMEL) when the trigger signal is active. In this case the lock diagnostics enable the action ACT. In this instance the trigger signal TRIGR must be active throughout the entire time. An error situation exists if the lock enabler UNLOCK does not appear within the time period, (lock not free). In this instance the action output ACT does not become active and the error output ERR is set. In addition, the logic at the UNLOCK input is analyzed and the error is entered in the error buffer. This error information is then displayed in a diagnostic signal on an attached MMI. This error message is terminated when the trigger signal TRIGR becomes inactive or the lock enabler UNLOCK becomes active.
The REACT input provides for the ACT output to be switched off or, respectively, prevents its activation without a locking error being reported.

Note: Please be sure that the REACT input is not negated. To switch off the action, REACT must have the value "1".

An active ACT output terminates the locking diagnostics and starts the action diagnostics.
The **action diagnostics** will be initiated when the defined ACT action becomes active. This action initiates an operation in the process, e.g. an output is set for putting the motor into standby operation. This operation has to trigger a specific reaction. This reaction mostly occurs with a set delay. However, if the reaction does not occur within the tolerance time DTIMEA, an error situation arises and the error output ERR becomes active. In addition, the logic at the UNLOCK input is analyzed and the error is entered in the error buffer. This error information is then displayed in a diagnostic signal on an attached MMI.
Monitoring is performed cyclically. Activation of the diagnostics and, at the same time, distribution of the cycle load can be achieved through the enable signal ED. The ED enable signal refers only to the activation of the diagnostics and has no effect on the ACT output of the locking diagnostics.
A positive edge of the ED enable signal (regardless at what time), or the locking signal UNLOCK becoming inactive while the TRIGR signal is active (in the action diagnostics phase), resets the function block and starts it in the locking diagnostics state.

Note: If in Concept in dialog **Project** → **Code generation options...** you select the option **Include diagnostics information**, the function block provides additional diagnostics codes which can be evaluated using diagnostics software. The function block, however, only makes the diagnostics codes available, if it is used in the FBD programming language.

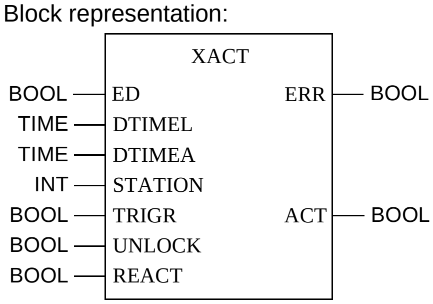
Note: NEVER use diagnostic EFBs in DFBs.

The parameters EN and ENO can additionally be projected.

Representation

Symbol

Block representation:



Parameter description

Block parameter description:

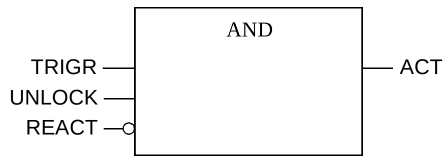
Parameters	Data type	Meaning
ED	BOOL	Enable diagnostics
DTIMEL	TIME	Tolerance time for locking diagnostics
DTIMEA	TIME	Tolerance time for action diagnostics
STATION	INT	Drop number (if no entry is made, drop number "0" will be used).
TRIGR	BOOL	Trigger signal
UNLOCK	BOOL	Lock
REACT	BOOL	Reaction input
ERR	BOOL	Error message; 0: no error; 1: Error
ACT	BOOL	Action output

Detailed description

Locking diagnostics parametering

Note: The ACT output is created with a logical AND from TRIGR and UNLOCK. REACT must not be active in this situation. Other inputs (e.g. ED) have no effect on this.

Representation of the relevant inputs for the ACT output



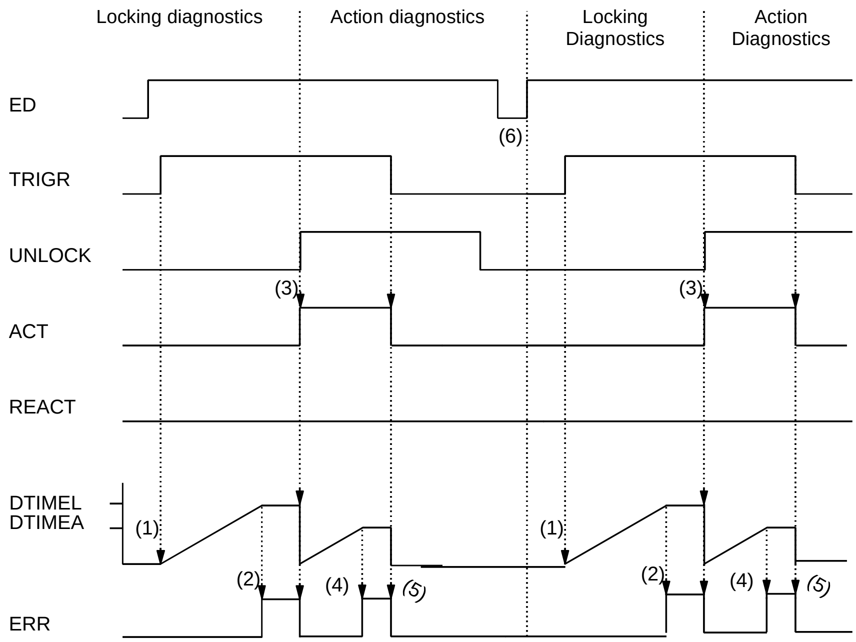
If the TRIGR input (trigger signal) becomes "1" and UNLOCK does not, the internal counter will be started.
When the default time at the DTIMEL input has expired, the ERR output will display an error; it remains active until TRIGR becomes "0", ACT becomes "1" or the diagnostics are deactivated.
If the trigger time (DTIMEL) is entered as "0", an error message is displayed as soon as an error situation occurs.
A detailed example showing the process of locking diagnostics can be found in the locking diagnostics timing diagram.
An active ACT output terminates the locking diagnostics and starts the action diagnostics.

Action diagnostics parametering

If the ACT output becomes "1" and REACT does not, the internal counter will be started.
If the action becomes inactive during processing, the monitoring time is stopped/reset or in the event of an error, the error processing is stopped.
When the default time at the DTIMEA input has expired, the ERR output will display an error; it remains active until ACT becomes "0", REACT becomes "1" or the diagnostics are deactivated.
If the tolerance time (DTIMEA) is entered as "0", an error message is displayed as soon as an error situation occurs.
An example for the process of lock / action diagnostics is given in the timing diagram.

Timing diagram

Locking / action diagnostics timing diagram



1. The internal time starts when TRIGR is "1" and UNLOCK is "0".
2. If the internal time reaches the DTIMEL value, an error will be reported.
3. If UNLOCK becomes "1", the error will be cancelled, the internal time is stopped/reset, and ACT becomes "1". By activating the action, the action diagnostics are switched. As the reaction has not yet occurred, the internal time is started.
4. If the internal time reaches the DTIMEA value, an error will be reported.
5. The error is cancelled and the internal time stopped/reset when ACT becomes "0".
6. When ED becomes "0", the function block is reset and locking diagnostics will start.

XLOCK: Extended locking diagnostics

12

Overview

At a Glance

This chapter describes the XLOCK block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The function block XLOCK is used for locking diagnostics and for enabling the action. Locking diagnostics are activated when the input with the TRIGR signals becomes active.
In control networks the trigger signal TRIGR (e.g. step counter, manual key) does not necessarily initiate the execution of an action directly, but is generally combined with locks from the process. It is therefore possible that the action ACT only becomes active after a time delay or not at all. It is the task of lock diagnostics to check whether UNLOCK is enabled within a tolerance time DTIME when the trigger signal is active. In this case the lock diagnostics enable the action ACT. In this instance the trigger signal TRIGR must be active throughout the entire time. An error situation exists if the lock enable UNLOCK does not appear within the time period, (lock not freed). In this instance the action output ACT does not become active and the error output ERR is set. In addition, the logic at the UNLOCK input is analyzed and the error is entered in the error buffer. This error information is then displayed in a diagnostic signal on an attached MMI. This error message is terminated when the trigger signal TRIGR is inactive or the lock enable UNLOCK becomes active. Contrary to the LOCK_DIA and XLOCK_DIA function blocks, the XLOCK function block has a REACT input that allows the ACT output to be switched off or prevents its activation, respectively, without a locking error being reported.

Note: Please be sure that the REACT input is not negated. To switch off the action, REACT must have the value "1".

The lock diagnostics terminate with an active action output ACT.
Monitoring is performed cyclically. Activation of the diagnostics and, at the same time, distribution of the cycle load can be achieved through the enable signal ED. The ED enable signal refers only to the activation of the diagnostics and has no effect on the ACT output.

Note: If, in Concept, in the dialog **Project** → **Code generation options...** you select the option **Include diagnostics information**, the function block provides additional diagnostics codes which can be evaluated using diagnostics software. The function block, however, only makes the diagnostics codes available, if used in the FBD programming language.

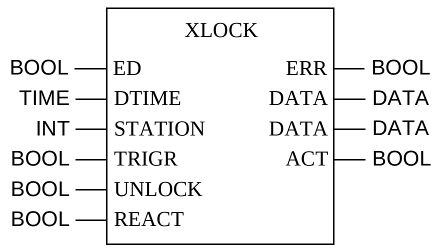
Note: NEVER use diagnostic EFBs in DFBs.

The parameters EN and ENO can additionally be projected.

Representation

Symbol

Block representation:



Parameter description

Block parameter description:

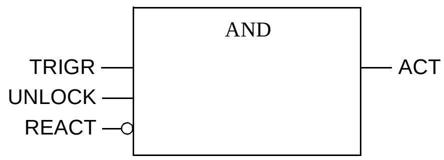
Parameters	Data type	Meaning
ED	BOOL	Enable diagnostics
DTIME	TIME	Tolerance time
STATION	INT	Drop number (if no entry is made, drop number "0" will be used).
TRIGR	BOOL	Trigger signal
UNLOCK	BOOL	Lock
REACT	BOOL	Reaction input
ERR	BOOL	Error message; 0: no error; 1: Error
ACT	BOOL	Action output

Detailed description

Parametering

Note: The ACT output is created with a logical AND from TRIGR and UNLOCK. REACT must not be active at that stage. Other inputs (e.g. ED) have no effect on this.

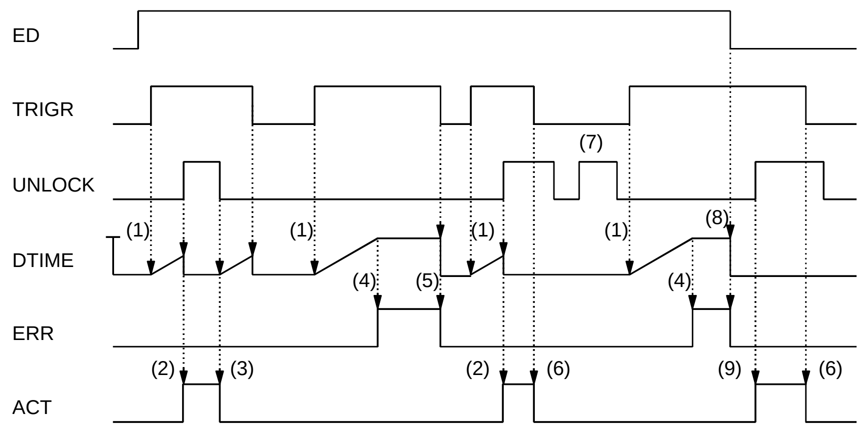
Representation of the relevant inputs for the ACT output



If the TRIGR input (trigger signal) becomes "1" and UNLOCK does not, the internal counter will be started.
When the default time at the DTIME input has expired, the ERR output will display an error; it remains active until TRIGR becomes "0", ACT becomes "1" or the diagnostics are deactivated.
If the trigger time (DTIME) is entered as "0", an error message is displayed as soon as an error situation occurs.
An example for the process of a lock diagnostic is given in the timing diagram.

Timing diagram

Locking diagnostics timing diagram



1. The internal time starts when TRIGR is "1" and UNLOCK is "0".
2. The internal time is stopped/reset and ACT becomes "1" when UNLOCK becomes "1".
3. ACT becomes "0" when UNLOCK becomes "0".
4. If the internal time reaches the DTIME value, an error will be reported.
5. The error is cancelled and the internal time stopped/reset when TRIGR becomes "0".
6. ACT becomes "0" when TRIGR becomes "0".
7. If UNLOCK is "1" and TRIGR is "0", the internal time does not start.
8. The error is cancelled and the internal time stopped/reset, if the enable signal ED becomes "0".
9. If TRIGR and UNLOCK are "1" and ED is "0", action becomes "1". ED has no effect on the ACT signal.
10. ACT becomes "0" when REACT becomes "1".
11. ACT becomes "1" when REACT becomes "0" and TRIGR and UNLOCK are "1".
12. ACT becomes "0" when REACT becomes "1" and TRIGR and UNLOCK are "1".
13. If UNLOCK is "0" and REACT is "1", ERR remains "0". (No error will be reported because there was a reaction to the action.)

XACT_DIA: Extended action diagnostics

13

Overview

At a Glance

This chapter describes the XACT_DIA block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The function block XACT_DIA is used for action diagnostics.
Action diagnostics are initiated when the defined action becomes active. This action initiates an operation in the process. This operation has to trigger a specific reaction. This reaction mostly occurs with a set delay. However, if the reaction does not occur within the tolerance time DTIME, an error situation arises and the error output ERR becomes active. In contrast to Locking diagnostics, in which the trigger of the diagnostics must remain active at all times, the behavior of the trigger (action) in action diagnostics can vary.
There are 3 different types of behavior:

- M behavior
- I behavior
- MI behavior

These possibilities vary in the behavior of the diagnostics if the action signal becomes "0" before an authorized value is placed at the reaction input.
The monitoring is performed cyclically. Activating the diagnostics, which results in a distribution of the cycle load, can be achieved through the enable signal "ED".

Note: If, in Concept, in the dialog **Project** → **Code generation options...** you activate the option **Include diagnostics information**, the function block provides additional diagnostics codes which can be evaluated using diagnostics software. However, the diagnostics codes are only made available from the function block if the function block is used in the FBD programming language.

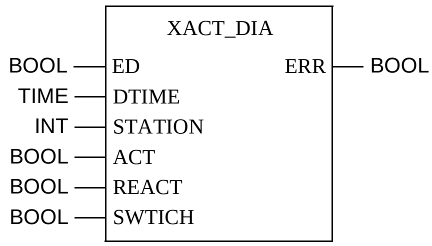
Note: NEVER use diagnostic EFBs in DFBs.

EN and ENO can be projected as additional parameters.

Representation

Symbol

Block representation:



Parameter description

Block parameter description:

Parameters	Data type	Meaning
ED	BOOL	Enable diagnostics
DTIME	TIME	Tolerance time
STATION	INT	Drop number (if no entry is made, drop number "0" will be used).
ACT	BOOL	Action signal
REACT	BOOL	Reaction signal
SWITCH	BOOL	M/I switch; 0: M behavior, 1: I behavior, 0/1: MI behavior
ERR	BOOL	Error message; 0: no error; 1: Error

Detailed description

Parametering

In order to control the different types of behavior (M, I, MI behavior), the appropriate value must be set at SWITCH and ACT.

Behavior	SWITCH	ACT
M behavior	0	0 or 1
I behavior	1	0 or 1
MI behavior	0 -> 1 (value modification within selected time)	1

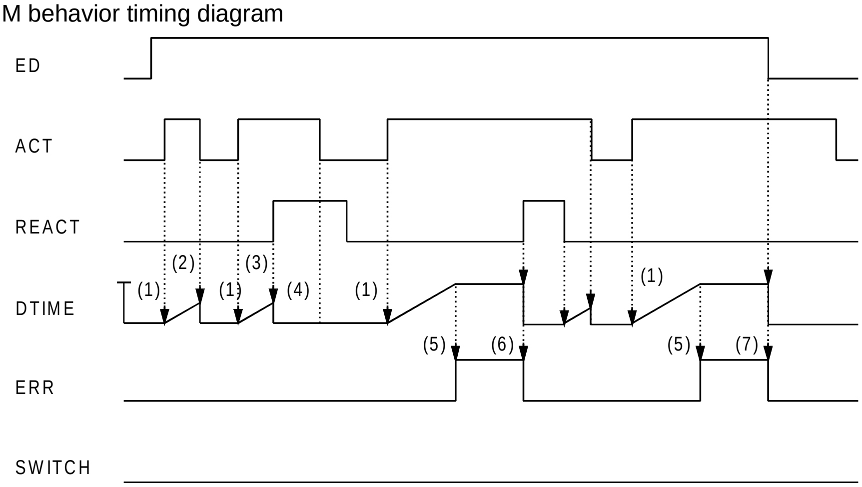
XACT_DIA: M behavior

Motor-like behavior

If the ACT input becomes "1" and REACT does not, the internal counter starts.
If the action becomes inactive during processing, the monitoring time is stopped/reset or in the event of an error, the error processing is stopped.
When the default time at the DTIME input has expired, the ERR output will display an error; it remains active until ACTION becomes "0", REACT becomes "1" or the diagnostics are deactivated.
If the tolerance time (DTIME) is entered as "0", an error message is displayed as soon as an error situation occurs.
An example for the process of action diagnostics with M behavior is given in the timing diagram.

Timing diagram

M behavior timing diagram



1. The internal time is started when ACT is "1" and REACT is "0".
2. The internal time is stopped/reset when ACT is "0".
3. The internal time is stopped/reset when REACT is "1".
4. Once the reaction has been detected, it is insignificant whether or not ACT is active.
5. If the internal time reaches the DTIME value, an error will be reported.
6. The error is cancelled and the internal time stopped/reset when REACT becomes "1".
7. The error is cancelled and the internal time stopped/reset, if the enable signal ED is "0".

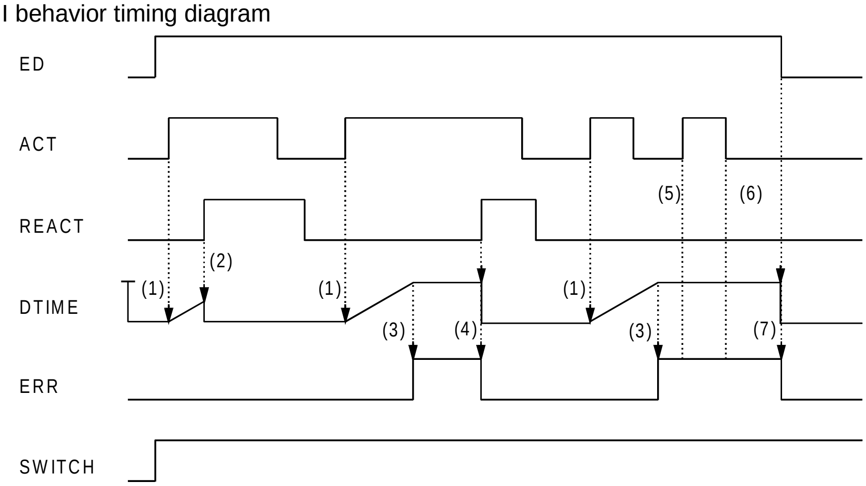
XACT_DIA: I behavior

Pulse behavior

After an edge of the action signal has been detected, diagnostics is activated and the monitoring time will start. The action signal's valency is no longer influential. When the default time at the DTIME input has expired, the ERR output will display an error; it remains active either until REACT becomes "1" or diagnostics is deactivated.
The diagnostics will only be terminated (in contrast to the case of M behavior) by the incoming defined reaction. In order to allow the diagnostics to be terminated despite any errors which may occur, the ED enable signal must be projected.
If the tolerance time (DTIME) is entered as "0", an error message is displayed as soon as an error situation arises.
An example for the process of action diagnostics with I behavior is given in the timing diagram.

Timing diagram

I behavior timing diagram



1. The internal time will start when ACT is "1" and REACT is "0".
2. The internal time is stopped/reset when REACT is "1".
3. If the internal time reaches the DTIME value, an error will be reported.
4. The error is cancelled and the internal time stopped/reset when REACT becomes "1".
5. If the action diagnostics are still being processed (e.g. error handling), an action positive edge is not significant.
6. If the action diagnostics are still being processed (e.g. error handling), an action negative edge is not significant.
7. The error is cancelled and the internal time stopped/reset, if the enable signal ED is "0".

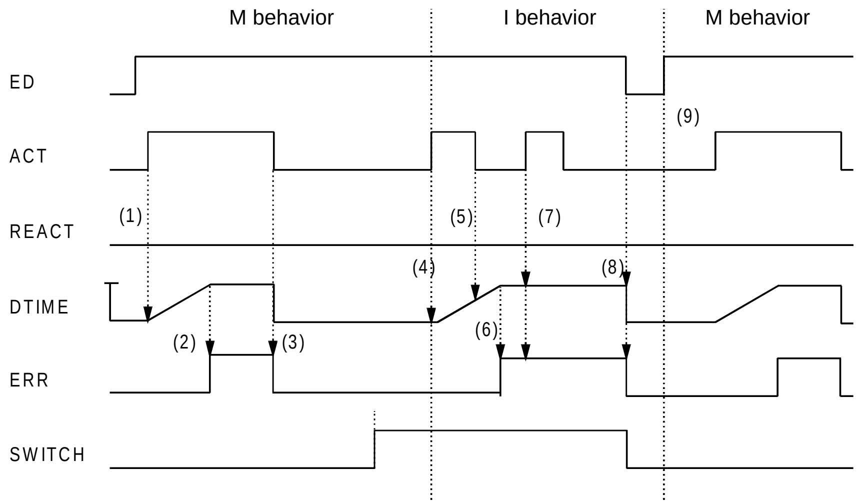
XACT_DIA: MI behavior

MI behavior

In the case of MI behavior, monitoring begins with M behavior. If during monitoring the ACT signal is "1" and the SWITCH signal becomes active (0 -> 1 edge), the diagnostics switch to I behavior
. This switch can only take place once and it is not possible to change back to M behavior during this monitoring cycle.
Since action diagnostics with I behavior can only be terminated via the defined reaction or the ED enable signal, the enable signal must also be projected in the case of MI behavior.
If the tolerance time (DTIME) is entered as "0", an error message is immediately displayed if an error situation occurs.
An example for the process of action diagnostics with MI behavior is given in the timing diagram.

Timing diagram

MI behavior timing diagram



1. The internal time will start when ACT is "1" and REACT is "0".
2. If the internal time reaches the DTIME value, an error will be reported.
3. With M behavior, the error will be cancelled, and the internal time stopped/reset when ACT becomes "0".
4. If SWITCH is "1" and ACT becomes "1", the diagnostics will switch from M behavior to I behavior. The internal time will also start when ACT is "1" and REACT is "0".
5. If the action diagnostics are still being processed (e.g. internal time started) during I behavior, a negative edge of the action is not significant.
6. If the internal time reaches the DTIME value, an error will be reported.
7. If the action diagnostics are still being processed (e.g. internal time started) during I behavior, a positive edge of the action is not significant.
8. The error is cancelled and the internal time stopped/reset, if the enable signal ED is "0".
9. If the enable signal ED returns to "1" or REACT becomes "1", a switch from I behavior to M behavior occurs.

XDYN_DIA: Extended dynamic diagnostics

14

Overview

At a Glance

This chapter describes the XDYN_DIA block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The function block XDYN_DIA is used for dynamic diagnostics.
For certain processes it is necessary to combine XLOCK_DIA (Extended locking diagnostics), XACT_DIA (Extended action diagnostics) and XREA_DIA (Extended reaction diagnostics) in one unit, which monitors the current state of the diagnostics. This is only possible via a special function block, which internally manages the current diagnostics status.
To prevent this function block from becoming too complicated, only one ED enable signal and one ERR error output have been defined.
The monitoring is performed cyclically. Activation of the diagnostics causing distribution of the cycle load can be achieved through the enable signal ED.

Note: If, in Concept, in the dialog **Project** → **Code generation options...** you select the option **Include diagnostics information**, the function block provides additional diagnostics codes which can be evaluated using diagnostics software. However, the function block only makes the diagnostics codes available if the function block is used in the FBD programming language.

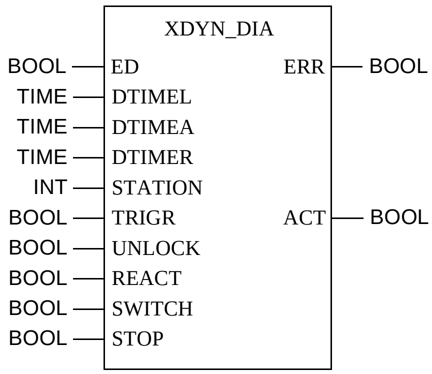
Note: NEVER use diagnostic EFBs in DFBs.

EN and ENO can be projected as additional parameters.

Representation

Symbol

Block representation:



Parameter description

Block parameter description:

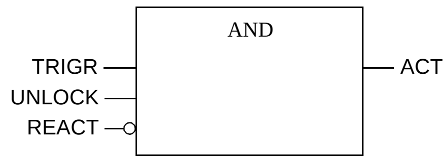
Parameters	**Data type**	**Meaning**
ED	BOOL	Enable diagnostics
DTIMEL	TIME	Tolerance time LOCK_DIA (locking diagnostics)
DTIMEA	TIME	Tolerance time ACT_DIA (action diagnostics)
DTIMER	TIME	Tolerance time REA_DIA (reaction diagnostics)
STATION	INT	Station number (if no entry is made, station number "0" will be used).
TRIGR	BOOL	Trigger
UNLOCK	BOOL	Lock
REACT	BOOL	Reaction signal
SWITCH	BOOL	M/I switch; 0: M behavior, 1: I behavior, 0/1: MI behavior
STOP	BOOL	Stop signal
ERR	BOOL	Error message; 0: No error; 1: Error
ACT	BOOL	Action enabling

Detailed description

Parametering

Note: The output is created with a logical AND from TRIGR and UNLOCK. Other inputs (e.g. ED) have no effect on this.

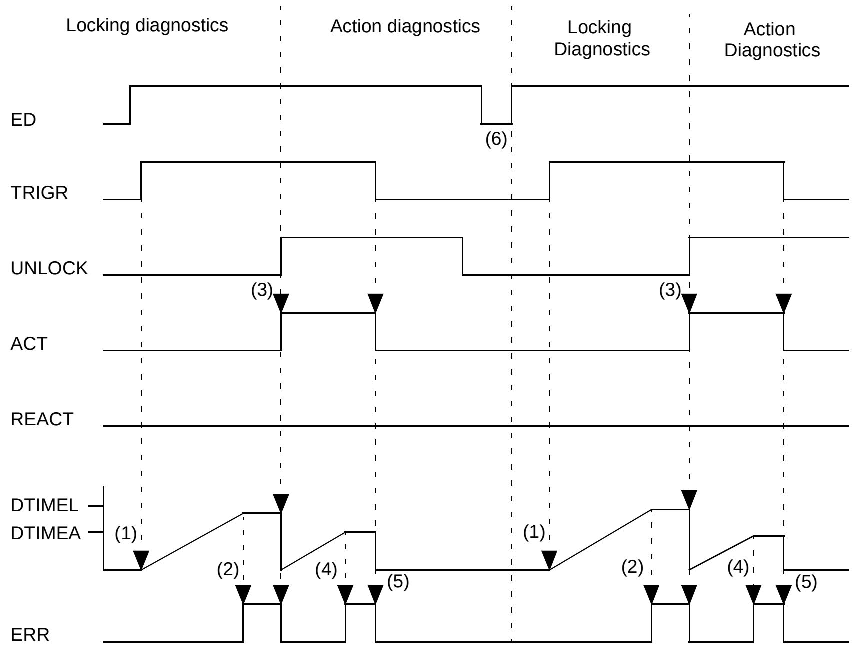
Representation of the relevant inputs for the ACT output



Parameterization for each diagnostics type can be found in the description for XLOCK_DIA, XACT_DIA and XREA_DIA.
An individual tolerance time (DTIMEL, DTIMEA, DTIMER) can be parametered for every diagnostics type.
An example for the process of a dynamic diagnostic is given in the timing diagram.

Timing diagram

Timing diagram for dynamic diagnostics



1. The internal time starts when TRIGR is "1" and UNLOCK is "0".
2. If the internal time reaches the DTIMEL value, an error will be reported.
3. If UNLOCK becomes "1", the error will be cancelled, the internal time is stopped/reset, and ACT becomes "1". The pass power of the action causes a switch to action diagnostics. As the reaction has still not occurred, the internal time is started.
4. If the internal time reaches the DTIMEA value, an error will be reported.
5. With M behavior, the error will be cancelled, and the internal time stopped/reset when ACT becomes "0".
6. If SWITCH becomes "1" and ACT is "1", a switch from M behavior to I behavior occurs. The internal time will also start when ACT is "1" and REACT is "0".
7. If the action diagnostics are still in progress (e.g. internal time started) during I behavior, a negative edge of the action is not significant.
8. If the internal time reaches the DTIMEA value, an error will be reported.
9. If REACT becomes "1", the internal time is stopped/reset. The pass power of the reaction causes a switch to reaction diagnostics.
10. The internal time will start when REACT becomes "0".
11. If the internal time reaches the DTIMER value, an error will be reported.

12. The error will be cancelled and the internal time is stopped/reset when STOP becomes "1". The pass power of the stop signal causes a switch back to locking diagnostics.

XGRP_DIA: Extended signal group monitoring

15

Overview

At a Glance

This chapter describes the XGRP_DIA block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The XGRP_DIA function block is used for signal group monitoring.
The monitoring is performed cyclically. Activating the diagnostics which causes the distribution of the cycle load, can be achieved through the enable signal "ED".

Note: If, in Concept, in the dialog **Project** → **Code generation options...** you activate the option **Include diagnostics information**, the function block provides additional diagnostics codes which can be evaluated using diagnostics software. However, the function block only makes the diagnostics codes available if the function block is used in the FBD programming language.

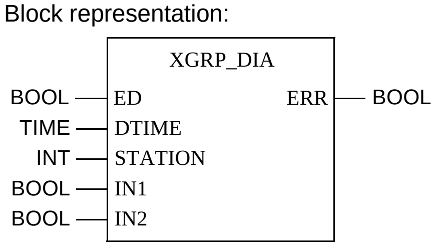
Note: NEVER use diagnostic EFBs in DFBs.

EN and ENO can be projected as additional parameters.

Representation

Symbol

Block representation:



Parameter description

Block parameter description:

Parameters	Data type	Meaning
ED	BOOL	Enable diagnostics
DTIME	TIME	Tolerance time
STATION	INT	Station number (if no entry is made, station number "0" will be used).
IN1	BOOL	1. Signal
IN2	BOOL	2. Signal
ERR	BOOL	Error message; 0: no error; 1:error

Detailed description

Parametering

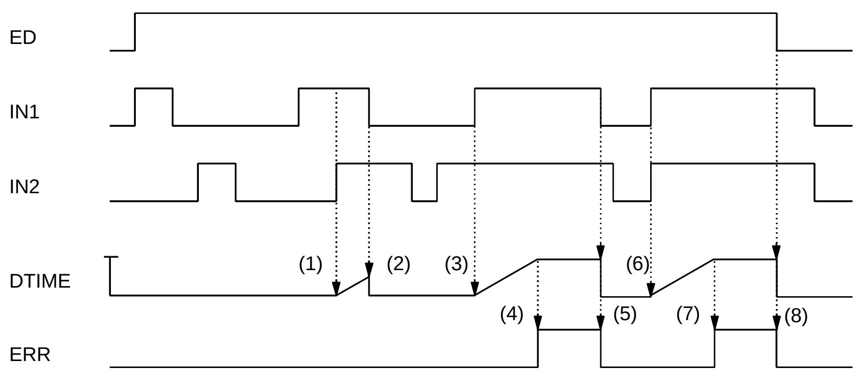
The inputs IN1 and IN2 are monitored to determined whether more than one input is "1".

Note: Unlike GRP_DIA, this function block only possesses two INx-inputs, as with the XGRP_DIA there is an additional analysis of the faulty signals and an entry in the error buffer. This analysis can only be made for 2 signals.

Deactivating the diagnostics or the attached correct values at the inputs will reset the internal counter to "0". When the default time at the DTIME input has expired, the ERR output displays an error; it remains active until fewer than two inputs are "1" or until the diagnostics is deactivated.
If a tolerance time (DTIME) of "0" is entered, an error message comes up immediately if more than one input becomes "1".
An example for the process of signal group monitoring is given in the timing diagram

Timing diagram

Signal group monitoring timing diagram



1. The internal time is started when IN1 and IN2 simultaneously become "1".
2. The internal time is stopped/reset when IN1 becomes "0".
3. The internal time is started when IN1 and IN2 become "1" simultaneously.
4. If the internal time reaches the DTIME value, an error will be reported.
5. The error is cancelled and the internal time is stopped/reset when IN1 becomes "0".
6. The internal time is started when IN1 and IN2 become"1"simultaneously.
7. If the internal time reaches the DTIME value, an error will be reported.
8. The error is cancelled and the internal time stopped/reset, if the enable signal ED is "0".

XLOCK_DIA: Extended locking diagnostics

16

Overview

At a Glance

This chapter describes the XLOCK_DIA block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The function block XLOCK_DIA is used for locking diagnostics and to enable the action.
Locking diagnostics are activated when the input with the TRIGR trigger signals becomes active.
In control networks the trigger signal TRIGR (e.g. step counter, manual key) does not necessarily initiate the execution of an action directly, but is generally combined with locks from the process. It is therefore possible that the action ACT only becomes active after a time delay or not at all. It is the task of lock diagnostics to check whether UNLOCK is enabled within a tolerance time DTIME when the trigger signal is active. In this case the lock diagnostics enables the action ACT. In this instance the trigger signal TRIGR must be active throughout the entire time An error situation exists if the lock enabler UNLOCK does not occur within the time period, (lock not free). In this instance the action output ACT does not become active and the error output ERR is set. This error message is terminated when the trigger signal TRIGR is inactive or the lock enabler UNLOCK becomes active.
The lock diagnostics is terminated with an active action output ACT.
The monitoring is carried out cyclically. The activation of the diagnostics which causes the distribution of the cycle load can be achieved through the enable signal ED. The ED enable signal refers only to the activation of the diagnostics and has no effect on the ACT output.

Note: Unlike the XLOCK_DIA function block, the XLOCK function block has a reaction input REACT, that enables the action output ACT to be switched off, i.e. hindering its activation, without a lock error being displayed.

Note: If, in Concept, in the dialog **Project** → **Code generation options...** you select the option **Include diagnostics information**, the function block provides additional diagnostics codes which can be evaluated using diagnostics software. However, the function block only makes diagnostics codes available if the function block is used in the FBD programming language.

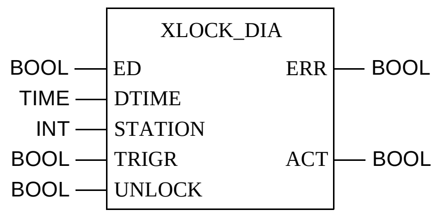
Note: NEVER use diagnostic EFBs in DFBs.

As additional parameters EN and ENO can be projected.

Representation

Symbol

Block representation:



Parameter description

Block parameter description:

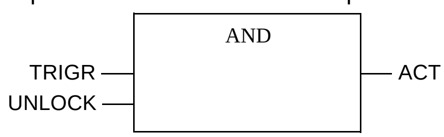
Parameters	Data type	Meaning
ED	BOOL	Enable diagnostics
DTIME	TIME	Tolerance time
STATION	INT	Station number (if no entry is made, station number "0" will be used).
TRIGR	BOOL	Trigger signal
UNLOCK	BOOL	Lock
ERR	BOOL	Error message; 0: no error; 1: Error
ACT	BOOL	Action output

Detailed description

Parametering

> **Note:** The output is created with a logical AND from TRIGR and UNLOCK. Other inputs (e.g. ED) have no effect on this.

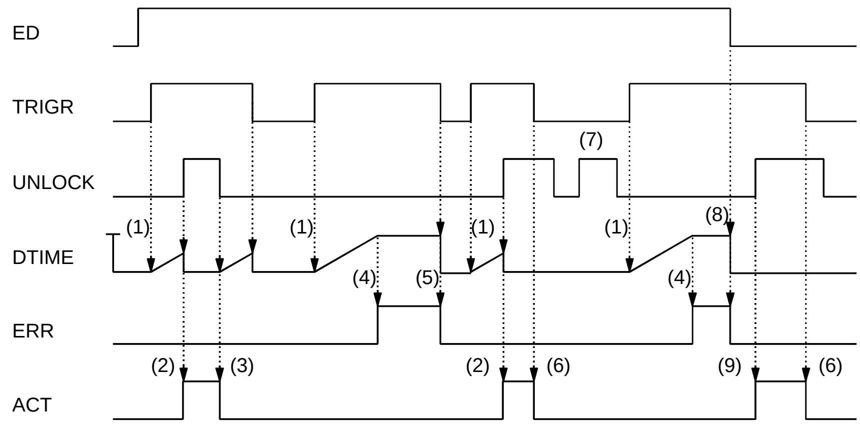
Representation of the relevant inputs for the ACT output



If the TRIGR input (trigger signal) becomes "1" and UNLOCK doesn't, the internal counter will be started.
When the default time at the DTIME input has expired, the ERR output will display an error; it remains active until TRIGR becomes "0", ACT becomes "1" or diagnostics is deactivated.
If the trigger time (DTIME) is entered as "0", an error message is displayed as soon as an error situation occurs.
An example for the process of a lock diagnostic is given in the timing diagram.

Timing diagram

Locking diagnostics timing diagram



1. The internal time starts when TRIGR is "1" and UNLOCK is "0".
2. The internal time is stopped/reset and ACT becomes "1" when UNLOCK becomes "1".
3. ACT becomes "0" when UNLOCK becomes "0".
4. If the internal time reaches the DTIME value, an error will be reported.
5. The error is cancelled and the internal time stopped/reset when TRIGR becomes "0".
6. ACT becomes "0" when TRIGR becomes "0".
7. If UNLOCK is "1" and TRIGR is "0", the internal time does not start.
8. The error is cancelled and the internal time stopped/reset, if the enable signal ED is "0".
9. If TRIGR and UNLOCK are "1" and ED is "0", action becomes "1". ED has no effect on the ACT signal.

XPRE_DIA: Extended process requirement monitoring

17

Overview

At a Glance

This chapter describes the XPRE_DIA block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The function block XPRE_DIA is used for the monitoring of process requirements. Process requirements or preconditions are process characteristics that are absolutely necessary for the operation of the machine or system (e.g. coolant, emergency stop). General requirements are for example requirements for machine operating modes or basic settings.
The absence of such requirements is monitored. The monitoring is carried out cyclically. The activation of the diagnostics which causes the distribution of the cycle load can be achieved through the enable signal ED.
The number of inputs IN can be increased up to a maximum of 30 by vertically modifying the size of the block.

Note: If, in Concept, in the dialog **Project** → **Code generation options...** you select the option **Include diagnostics information**, the function block provides additional diagnostics codes which can be evaluated using diagnostics software. However, the function block only makes the diagnostics codes available if the function block is used in the FBD programming language.

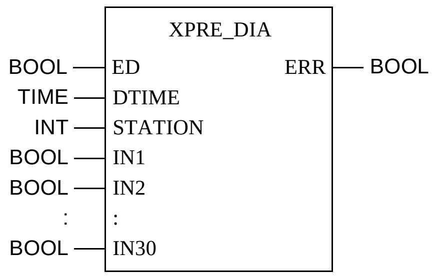
Note: NEVER use diagnostic EFBs in DFBs.

As additional parameters EN and ENO can be projected.

Representation

Symbol

Block representation:



Parameter description

Block parameter description:

Parameters	**Data type**	**Meaning**
ED	BOOL	Enable diagnostics
DTIME	TIME	Tolerance time
STATION	INT	Station number (if no entry is made, station number "0" will be used).
IN1	BOOL	1. Process requirement
:	:	:
IN30	BOOL	30. Process requirement
ERR	BOOL	Error message; 0: no error; 1: Error

Detailed description

Parametering

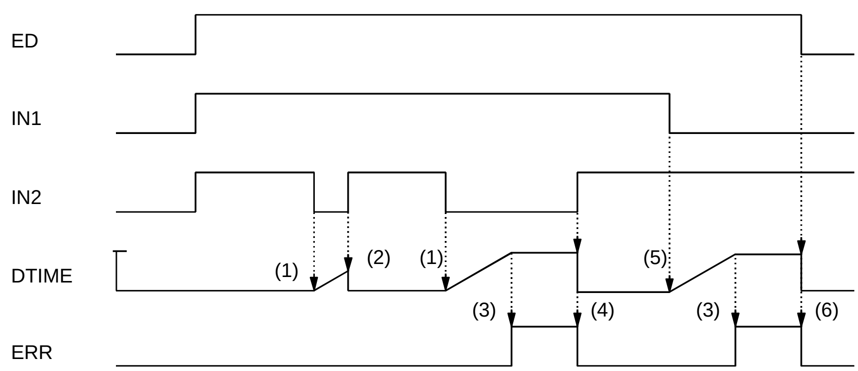
If at least one of the signals connected to INx becomes "0" and the diagnostics are active, the internal counter will be started.

> **Note:** Please note that all visible and unlinked inputs are automatically assigned a "0", i.e. create only as many IN inputs as are actually needed.

The deactivation of the diagnostics or of the attachment of the correct input value stops the counter (the requirements may be contain errors during the tolerance time) and sets the counter back to "0".
When the default time at the DTIME input has expired, the ERR output displays an error; it remains active until the requirements are "1" or the diagnostics are deactivated.
If the tolerance time entered is DTIME "0", there is an immediate error message when the static conditional values (INx) become "0".
An example for process requirement monitoring is given in timing diagram.

Timing diagram

Monitoring of process requirements timing diagram



1. The internal time will start when IN2 becomes "0".
2. The internal time is stopped/reset when IN2 becomes "1".
3. If the internal time reaches the DTIME value, an error will be reported.
4. The error is cancelled and the internal time is stopped/reset when IN2 becomes "1".
5. The internal time will start when IN1 becomes "0".
6. The error is cancelled and the internal time stopped/reset, if the enable signal ED is "0".

XREA_DIA: Extended reaction diagnostics

18

Overview

At a Glance

This chapter describes the XREA_DIA block.

What's in this Chapter?

This Chapter contains the following Maps:

Brief description

Function description

The function block REA_DIA is used for reaction diagnostics.
Once the expected reaction has occurred in the Actions diagnostics, the reaction diagnostics are checked to ascertain whether the process contains the status.
The process reaction, defined as a term or a signal, is checked through the reaction diagnostics to determine whether the status is stable During technical processes it can occur that the reactions change momentarily (e.g. hitting limit switches). In order for the reaction diagnostics not to activate the error message ERR directly in such a case, a tolerance time DTIME can be defined. An error signal occurs if this time is exceeded. The error signal becomes inactive when the reaction returns to the setpoint status or when the stop condition is met.
The stop condition terminates reaction diagnostics.
The monitoring is carried out cyclically. The activation of the diagnostics and thereby the distribution of the cycle load can be achieved through the enable signal ED.

Note: If, in Concept, in the dialog **Project → Code generation options...** you select the option **Include diagnostics information**, the function block provides additional diagnostics codes which can be evaluated using diagnostics software. However, the function block only makes the diagnostics codes available if the function block is used in the FBD programming language.

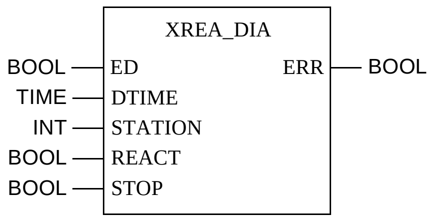
Note: NEVER use diagnostic EFBs in DFBs.

As additional parameters EN and ENO can be projected.

Representation

Symbol

Block representation:



Parameter description

Block parameter description:

Parameters	Data type	Meaning
ED	BOOL	Enable diagnostics
DTIME	TIME	Tolerance time
STATION	INT	Drop number (if no entry is made, drop number "0" will be used).
TRIG	BOOL	Trigger
REACT	BOOL	Reaction signal
STOP	BOOL	Stop signal
ERR	BOOL	Error message; 0: no error; 1: Error

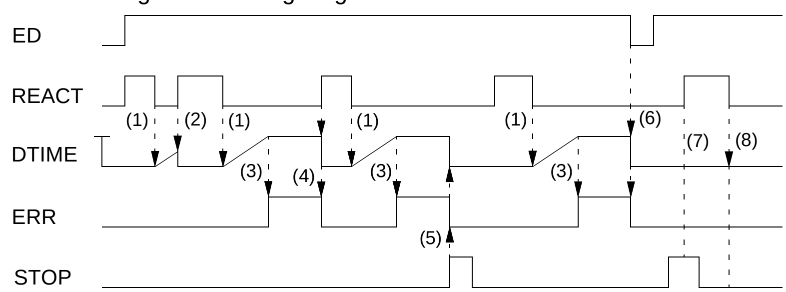
Detailed description

Parametering

If the REACT input becomes "0", the internal counter will be started.
When the default time at the DTIME input has expired, the ERR output will display an error; it remains active until REACT becomes "1", STOP becomes "1" or the diagnostics are deactivated.
If the tolerance time (DTIME) is entered as "0", an error message is displayed as soon as an error situation occurs.
The timing diagram provides an example for the process of a reaction diagnostics

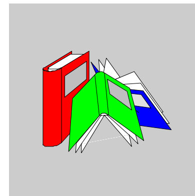
Timing diagram

Reaction diagnostics timing diagram



1. The internal time will start when REACT becomes "0".
2. The internal time is stopped/reset when REACT is "1".
3. If the internal time reaches the DTIME value, an error will be reported.
4. The error is cancelled and the internal time stopped/reset when REACT becomes "1".
5. The error will be cancelled and the internal time is stopped/reset when STOP becomes "1".
6. The error is cancelled and the internal time stopped/reset, if the enable signal ED is "0".
7. If REACT becomes "1", when STOP is "1", the reaction diagnostics are not started.
8. If REACT subsequently becomes "0", the internal time is not started, even if STOP is "0" again.

Glossary



A

Action signals Action signals represent the states of the action outputs assigned in the application program (e.g. "motor moving" signal).

Active window The window that is currently selected. Only one window can be active at a given point in time. If a window becomes active, the colour of its title bar changes to differentiate it from the other windows. Windows not selected are inactive.

Addresses (Direct) addresses are areas of memory in the PLC. These are in the signal memory and can be allocated to input/output modules.

Application window The window that contains the workarea, the menu bar and the tool bar for the application program. The name of the application program appears in the title bar. An application window can contain several document windows.
In the Handtableau, the application window corresponds to a project.

Atrium The PC-based controller is fitted to a standard AT circuit board and can be operated in a host computer in an ISA bus slot. The module has a motherboard (requires SA85 driver) with two slots for PC104 daughter boards. Of these, one PC104 daughter board is used for the CPU and the other for the control of the INTERBUS.

B

Base 16 literals

Base 16 literals are used to represent integer values in the hexadecimal system. The base must be identified by means of the prefix 16#. The values are not permitted to have a sign (+/-). Individual underscore characters (_) between the digits are not significant.

Example
16#F_F or 16#FF (decimal 255)
16#E_0 or 16#E0 (decimal 224)

Base 2 literals

Base 2 literals are used to represent integer values in the binary system. The base must be identified by means of the prefix 2#. The values are not permitted to have a sign (+/-). Individual underscore characters (_) between the digits are not significant.

Example
2#1111_1111 or 2#11111111 (decimal 255)
2#1110_0000 or 2#11100000 (decimal 224)

Base 8 literals

Base 8 literals are used to represent integer values in the octal system. The base must be identified by means of the prefix 8#. The values are not permitted to have a sign (+/-). Individual underscore characters (_) between the digits are not significant.

Example
8#3_77 or 8#377 (decimal 255)
8#34_0 or 8#340 (decimal 224)

Basic functions/ function blocks (EFB)

Term for functions or function blocks for which the type definitions are not defined in one of the IEC languages, i.e. their bodies, e.g., cannot be modified using the DFB editor (Concept-DFB). EFB types are programmed in "C" and are provided in pre-compiled form in libraries.

BOOL

BOOL stands for the data type "boolean". The length of the data elements is 1 bit (stored in 1 byte in the memory). The range of values for this type of data is 0 (FALSE) and 1 (TRUE).

BYTE

BYTE stands for the data type "sequence of 8 bits". The entry can be a base 2 literal, base 8 literal or base 16 literal. The length of the data elements is 8 bits. A range of numerical values cannot be assigned to this data type.

C

Call The process that is initiated by the execution of an operation.

Constants Constants are unlocated variables to which a value is assigned that cannot be changed by the program logic (write-protected).

Current parameter Currently connected input-/output parameter.

D

Data types The overview shows the hierarchy of the data types, as they are used for inputs and outputs for functions and function blocks. Generic data types are identified by the prefix "ANY".

- ANY_ELEM
  - ANY_NUM
    ANY_REAL (REAL)
    ANY_INT (DINT, INT, UDINT, UINT)
  - ANY_BIT (BOOL, BYTE, WORD)
  - TIME
- System data types (IEC extensions)
- Derived (from 'ANY' data types)

Defined literals If you want to define the data type for a literal yourself, you can do this using the following construction: 'data type name'#'value of the literal'.

Example
INT#15 (data type: integer, value: 15),
BYTE#00001111 (data type: byte, value: 00001111)
REAL#23 (data type: real, value: 23)

For the allocation of the REAL data type, it is also possible to enter the value in the following way: 23.0.
If a decimal place is entered, the REAL data type is automatically assigned.

Derived data type Derived data types are data types that have been derived from the basic data types and/or other derived data types. Derived data types are defined in the Concept data type editor. A differentiation is made between global data types and local data types.

Derived Function Block (DFB)

A derived function block represents the call for a derived function block type. You will find details on the graphic form of the call in the definition "function block (instance)". Contrary to calls for EFB types, calls for DFB types are marked with double vertical lines on the left and right sides of the square block symbol.
The body of a derived function block type is written in FBD language, however only in the current version of the programming system. Other IEC languages can currently not be used for the definition of DFB types, derived functions can also not be defined yet in the current version.
A differentiation is made between local and global DFBs.

DINT

DINT stands for the data type "double integer". The entry can be an integer literal, base 2 literal, base 8 literal or base 16 literal. The length of the data elements is 32 bits. The range of values for this data type is from -2 exp (31) to 2 exp (31) -1.

Direct representation

A method for representing variables in the PLC program from which the allocation to the logical memory location can be derived directly and, indirectly, the physical memory location.

Document window

A window within an application window. Several documents can be open simultaneously in an application window. However only one document window can be active. Document windows in Concept are, e.g., sections, the message window, the reference data editor and the PLC configuration.

Duration literal

Permitted units for durations (TIME) are days (D), hours (H), minutes (M), seconds (S) and milliseconds (MS) or combinations thereof. The duration must be identified by the prefix t#, T#, time# or TIME#. The "Overflow" of the maximum value unit is permitted; e.g. the entry T#25H15M is permitted.

Example
t#14MS, T#14.7S, time#18M, TIME#19.9H, t#20.4D, T#25H15M, time#5D14H12M18S3.5MS

E

EN / ENO (enable / fault signalling)
If the value of EN is equal to "0", when the FFB is called the algorithms that are defined by the FFB are not carried out and all outputs retain their previous value. The value of ENO is automatically set to "0" in this case. If the value of EN is equal to "1", when the FFB is called the algorithms that are defined by the FFB are carried out. Following the error-free execution of these algorithms the value of ENO is automatically set to "1". If an error occurs during the execution of these algorithms, ENO is automatically set to "0" . The output behaviour of the FFBs depends on whether the FFBs are called without EN/ENO or with EN=1. If the indication of EN/ENO is enabled, it is imperative that the EN input is connected. Otherwise the FFB is never carried out. The projection of EN and ENO is enabled or disabled in the function block properties dialog box. The dialog box is opened using the menu commands **Objects** → **Properties...** or by double clicking the FFB.

Equipment
A unit/device, e.g. pump, lifter or motor is termed an item of equipment. The equipment forms the central projection element. It is characterised by the configuration (manual/switch flag), actions, and reactions. Equipment can be combined into equipment groups.

Equipment group
The individual items of equipment configured are combined into an equipment group based on logical or technical considerations. This facilitates technical or machine-orientated representation of the equipment at Handtableau program runtime. In addition, the connection of the individual items of equipment during program runtime is performed via the selection of the equipment groups.

Error
If an error is detected during the processing of an FFB (e.g. incorrect input values or a timing error), an error message is displayed; you can view the error using the menu command **Online** → **Event Display...**. In FFBs the ENO output is set to "0".

Evaluation
The process by means of which a value for a function or for the outputs of a function block is determined during program execution.

F

FFB (Functions/ Function Blocks)
Collective term for EFB (basic functions/function blocks) and DFB (derived function blocks)

Field variables
Variables to which a defined derived data type is allocated with the aid of the keyword ARRAY (field). A field is a collection of data elements of the same data type.

Function (FUNK)
A program organisation unit that on execution provides exactly one data element. A function has no internal status information. Multiple calls of the same function with the same input parameter values always produce the same output values.
You will find details on the graphic form of function calls in the definition "function block (instance)". Contrary to function block calls, function calls have only one unnamed output, as its name is the same as the name of the function itself. In FBD each call is identified by a unique number via the graphic block; this number is generated automatically and cannot be changed.

Function block (instance) (FB)
A function block is a program organisation unit that calculates values for its outputs and internal variable(s) in accordance with the functionality defined in its function block type description, when it is called as a specific instance. All values for the outputs and internal variables of a specific function block instance are retained from one call of the function block to the next. Multiple calls of the same function block instance with the same arguments (values for input parameters) do not therefore necessarily produce the same output value(s).
Each function block instance is represented graphically using a rectangular symbol. The name of the function block type is given in the centre of the top of the rectangle. The name of the function block instance is also given at the top, however outside of the rectangle. This is automatically generated on the creation of an instance, however it can be modified by the user as required. Inputs are displayed on the left, outputs on the right of the block. The names of the formal input/output parameters are displayed inside the square at the appropriate point.
The above description of the graphic representation is in principle also valid for function calls and for DFB calls. Differences are described in the corresponding definitions.

Function block language (FBD)
One or more sections that contain the graphically displayed network of functions, function blocks and links.

Function block type
A language element comprising: 1. The definition of a data structure divided into input, output, and internal variables; 2. A set of operations that are performed with the elements in the data structure when an instance of the function block type is called. This set of operations can be formulated either in one of the IEC languages (DFB type) or in "C" (EFB type). One function block type can be multiply instanced (called).

Function counter The function counter is used to uniquely identify a function in a program or DFB. The function counter cannot be edited and is assigned automatically. The function counter always has the structure: .n.m

n = number of the section (sequential number)
m = number of the FFB object in the section (sequential number)

G

Generic literals If it is unimportant to you which data type a literal has, simply enter the value for the literal. In this case, Concept automatically assigns a suitable data type.

Global derived data types Global derived data types are available in each Concept project and are saved in the DFB directory directly under the Concept directory.

Global DFBs Global DFBs are available in each Concept project and are saved in the DFB directory under the Concept directory.

Groups (EFBs) Some EFB libraries (e.g. the IEC library) are subdivided into groups. This makes it considerably easier to locate EFBs in large libraries.

I

I/O installation list In the I/O installation list the I/O and expert modules of the different central units are configured.

Icon Graphic representation of different objects in Windows, e.g. drives, application programs and document windows.

IEC 1131-3 International standard: Programmable Controllers - Part 3: Programming Languages. March 1993.

IEC naming convention (identifier) An identifier is a sequence of letters, digits and underscores that must start with a letter or an underscore (e.g. name of the function block type, an instance, a variable or a section). Letters from national character sets (e.g.: ö,ü, é, õ) can be used, except in project and DFB names.

Underscore characters are significant in identifiers; e.g. "A_BCD" and "AB_CD" are interpreted as different identifiers. Several leading underscore characters and sequences of several underscore characters are not allowed.
Identifiers must not contain any spaces. Upper and lower case is not significant; e.g.. "ABCD" and "abcd" are interpreted as the same identifier.
Identifiers are not allowed to be keywords.

Initial value
The value assigned to a variable at program start. The value is assigned in the form of a literal.

Input bits (1x reference)
The 1/0 state of input bits is controlled by the process data that passes from an input device to the CPU.

> **Note:** The x that follows the first digit of the reference type represents a five-digit memory location in the user data memory, e.g. the reference 100201 signifies an input bit at address 201 of the signal memory.

Input parameter (input)
Passes the associated argument during a FFB call.

Input words (3x references)
An input word contains information that stems from an external source and is represented by a 16 bit number. A 3x register can also contain 16 sequential input bits that have been read into the register in binary or BCD (binary coded decimal) format. Note: The x that follows the first digit of the reference type represents a five-digit memory location in the user data memory, e.g. the reference 300201 signifies a 16-bit input word at address 201 of the signal memory.

Instance name
An identifier that belongs to a specific function block instance. The instance name is used to uniquely identify a function block in a program organisation unit. The instance name is generated automatically, however it can be edited. The instance name must be unique in the entire program organisation unit; here a differentiation is not made between upper and lower case. If the name entered already exists, you will be warned and must choose a different name. The instance name must comply with the IEC naming conventions, otherwise an error message is displayed. The automatically generated instance name always has the structure: FBI_n_m

FBI = function block instance
n = number of the section (sequential number)
m = number of the FFB object in the section (sequential number)

Instancing
The generation of an instance.

INT
INT stands for the data type "integer". The entry can be an integer literal, base 2 literal, base 8 literal or base 16 literal. The length of the data elements is 16 bits. The range of values for this data type is from -2 exp (15) to 2 exp (15) -1.

Integer literals Integer literals are used for entry of integer values in the decimal system. The values can have a sign (+/-) in front. Individual underscore characters (_) between the digits are not significant.

Example
-12, 0, 123_456, +986

K

Keywords Keywords are unique combinations of characters that are used as special syntactical elements as defined in the Appendix B of IEC 1131-3. All keywords that can be used in the IEC 1131-3 and thus in Concept are listed in Appendix C of IEC 1131-3. These listed keywords are not permitted to be used for any other purpose, e.g., not as variable names, section names, instance names, etc.

L

Language element Each basic element in one of the IEC programming languages, e.g. a step in SFC, a function block instance in FBD or the initial value of a variable.

Library Collection of software objects that are intended for reuse during the programming of new projects, or even for the creation of new libraries. Examples are the library for the basic function block types.
EFB libraries can be subdivided into groups.

Link A control or data flow link between graphic objects (e.g. function blocks in the FBD editor) within a section, represented graphically as a line.

Literals Literals are used to provide inputs on FFBs, transition conditions, etc directly with values. These values cannot be overwritten by the program logic (write protected). Here a differentiation is made between generic and classified literals.
In addition, literals are used to assign a value to a constant or an initial value to a variable.
Entry is made as base 2 literal, base 8 literal, base 16 literal, integer literal, real literal or real literal with exponent.

Local derived data types Local derived data types are only available in a single Concept project and its local DFBs and are stored in the DFB directory under the project directory.

Local DFBs

Local DFBs are only available in a single Concept project and are saved in the DFB directory under the project directory.

Located variable

Located variables are assigned to a memory address (reference addresses 0x, 1x, 3x,4x). The value of this variable is saved in the signal memory and can be changed online using the reference data editor. These variables can be addressed using their symbolic names or using their reference address.

All inputs and outputs for the PLC are connected to the signal memory. The access of the program to peripheral signals that are connected to the PLC is only performed using located variables. External accesses via Modbus or Modbus Plus interfaces on the PLC, e.g., from information display systems, are also possible via located variables.

M

Manual flag

If the manual control variable is configured as a manual flag, the variable is set to "1" with the rising edge of the signal as long as the corresponding control key is pressed.

Multielement variables

Variables to which a derived data type defined using a STRUCT or ARRAY is assigned.
A differentiation is made here between field variables and structured variables.

O

OFS (OPC Factory Server)

The OPC Factory Server (OFS) is the interface between the individual software components for the display of information, Handtableau, programming and application program. A prerequisite for the use of this interface is that all users are capable of communicating using OPC (OLE for Process Control). The OFS runs in the background.

Output parameter (output)

A parameter with which the result(s) of the evaluation of a FFB is/are fed back.

Output/flag bits (0x references)
An output/flag bit can be used to control real output data using an output module in the control system, or to define one or more discrete outputs in the signal memory. Note: The x that follows the first digit of the reference type represents a five-digit memory location in the user data memory, e.g. the reference 000201 signifies an output or flag bit at address 201 of the signal memory.

Output/flag words (4x references)
An output/flag word can be used for the storage of numerical data (binary or decimal) in the signal memory, or also for the transmission of data from the CPU to an output module in the control system. Note: The x that follows the first digit of the reference type represents a five-digit memory location in the user data memory, e.g. the reference 400201 signifies a 16-bit output/flag word at address 201 of the signal memory.

P

PLC
Programmable controller

Program
The topmost program organisation unit. A complete program is loaded on a single PLC.

Program cycle
A program cycle comprises the reading of the inputs, the processing of the program logic and the output of the outputs.

Program organisation unit
A function, a function block, or a program. This term can relate to either a type or an instance.

Programming unit
Hardware and software that supports programming, projecting, testing, commissioning and faultfinding in PLC applications, as well as in decentral system applications to facilitate source documentation and archiving. The programming unit can, amongst other tasks, be used for the display of process information.

Project
General term for the topmost level of a software tree structure that defines the superordinate project name of a PLC application. After definition of the project name, you can save your system configuration and your control program under this name. All data that is produced during the creation of the configuration and the program belong to this superordinate project for this special automation task. General term for the complete set of programming and projecting information in the project database, this represents the source code that describes the automation of a system.

Project database The database in the programming unit that contains the projection information for a project.

R

Reaction signals Reaction signals represent the states of the process reactions assigned and additional signals such as, e.g., interlocks or limit switches.

REAL REAL stands for the data type "floating point number". The entry is made as a real literal or as a real literal with exponent. The length of the data elements is 32 bits. The range of values for variables of this data type is from 8.43E-37 to 3.36E+38.

Real literals Real literals are used for entry of floating point values in the decimal system. Real literals are identified by the entry of the decimal point. The values can have a sign (+/-) in front. Individual underscore characters (_) between the digits are not significant.

Example
-12.0, 0.0, +0.456, 3.14159_26

Real literals with exponent Real literals with exponent are used for entry of floating point values in the decimal system. Real literals with exponent are identified by the entry of the decimal point. The exponent defines the power of ten with which the preceding number is to be multiplied to obtain the value to be represented. The values can have a sign (+/-) in front. Individual underscore characters (_) between the digits are not significant.

Example
-1.34E-12 or -1.34e-12
1.0E+6 or 1.0e+6
1.234E6 or 1.234e6

Reference Each direct address is a reference that starts with a code that defines whether the address is an input or output, and whether the data is a bit or word. References that start with the code 6 represent registers in the extended memory of the signal memory.
0x range = output/flag bits
1x range = input bits
3x range = input words
4x range = output/flag words
6x range = register in the extended memory

Note: The x that follows the first digit of each reference type represents a five-digit memory location in the user data memory, e.g. the reference 400201 signifies a 16-bit output or flag word at address 201 of the signal memory.

Register in the extended memory (6x reference)
6x references are flag words in the extended memory of the PLC. They can only be used for LL984 application programs and only if a CPU 213 04 or CPU 424 02 is used.

Runtime errors
Errors that occur during the processing of a program in the PLC, in SFC objects (e.g. steps) or FFBs. These are, e.g., value range overflows for numbers or timing errors for steps.

S

Section
A section can, e.g., be used to describe the method of operation of a technical unit, such as a motor.
A program or DFB comprises one or more sections. Sections can be programmed using the IEC programming languages FBD and SFC. Within a section only one of the stated programming languages is permitted to be used.
Each section has its own document window in Concept. However, for reasons of clarity, it is sensible to divide a very large section into several smaller sections. The scroll bar is used to scroll within the section.

Signal memory
The signal memory is the memory area for all parameters that are addressed in the application program using references (direct representation). For example, input bits, output/flag bits, input words, and output/flag words are contained in the signal memory.

Structured variable
Variables to which a derived data type defined using a STRUCT (structure) is assigned.
A structure is a collection of data elements with, in general, different data types (basic data types and/or derived data types).

Switch flag
If the manual control variable is configured as a switch flag, the variable is switched high (to "1") on the operation of the left control key, and switched low (to "0") on the operation of the right control key. The changeover takes place with the rising edge of the signal.

T

TIME

TIME stands for the data type "duration". The entry is made as a duration literal. The length of the data elements is 32 bits. The range of values for variables of this data type is from 0 to 2exp(32)-1. The unit for the data type TIME is 1 ms.

U

UDEFB

User defined basic functions/function blocks
Functions or function blocks that have been created in the programming language C and that Concept makes available in libraries.

UDINT

UDINT stands for the data type "unsigned double integer". The entry can be an integer literal, base 2 literal, base 8 literal or base 16 literal. The length of the data elements is 32 bits. The range of values for variables of this type is from 0 to 2exp(32)-1.

UINT

UINT stands for the data type "unsigned integer". The entry can be an integer literal, base 2 literal, base 8 literal or base 16 literal. The length of the data elements is 16 bits. The range of values for variables of this type is from 0 to (2exp16)-1.

Unlocated variable

An unlocated variable is not assigned a signal memory address. In this way they do not occupy any signal memory addresses. The value of this variable is saved internally in the system and can be changed using the reference data editor. These variables are only addressed using their symbolic names.

Signals that do not require direct access to the peripherals, e.g. intermediate results, system flags, etc., should preferably be declared as unlocated variables.

V

Variables

Variables are used for the exchange of data within sections, between several sections and between the program and the PLC.
Variables comprise at least one variable name and a data type.

If a variable is assigned a direct address (reference), the term located variable is used. If a variable is not assigned a direct address, the term unlocated variable is used. If a derived data type is assigned to the variable, the term multielement variable is used.
In addition, there are also constants and literals.

W

Warning
If a critical state is detected during the processing of a FFB or step (e.g. critical input values or time limit exceeded), a warning is displayed that you can view using the menu command **Online** → **Event display...**. In FFBs the ENO output remains at "1".

WORD
WORD stands for the data type "sequence of 16 bits". The entry can be a base 2 literal, base 8 literal or base 16 literal. The length of the data elements is 16 bits. A range of numerical values cannot be assigned to this data type.

Index