PL7 Junior/Pro

Setup
Diagnostic Functions

07/2008 eng

35005145.02

www.schneider-electric.com

Document Set

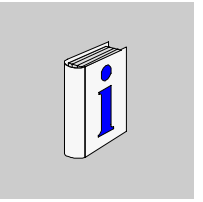
Introduction

This manual refers to the following documents:

- PL7 reference manual
- Runtime screen manual

Table of Contents

About the Book



At a Glance

Document Scope This manual deals with setting up the diagnostics functions on Premium PLCs using PL7 software.

Validity Note The update of this publication takes into account the functions of PL7V4.5. Nevertheless it can be used to set up earlier versions of PL7.

User Comments We welcome your comments about this document. You can reach us by e-mail at techpub@schneider-electric.com

General Introduction to Diagnostics Functions

1

Introduction

Subject of this chapter

This chapter describes the diagnostics package and its implementation in PL7.

What's in this Chapter?

This chapter contains the following topics:

Introduction to the diagnostics package

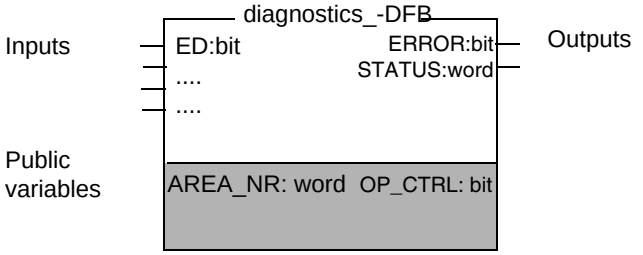
Diagnostics package

The Diagnostics Premium package is made up of:

- DFB diagnostics function blocks
- a diagnostics system
- an error message display system called Viewer

DFB diagnostics

DFB diagnostics are user function blocks programmed to carry out diagnostics functions of system or application types:



They feature directly providing error messages for a display system.

Diagnostics system

These diagnostics are carried out automatically. When the PLC processor detects system errors (for example: the watchdog is bypassed, input/output errors, division by zero), it transmits an error message to the display system

Viewer

The display window (called Viewer) built into the PL7 software, allows simple viewing of diagnostics messages:

Acknowlegement: 1/2	Error	Zone	Appearance: 6	Disappearence: 4	Message	Status 0 & Status 1
No Acknowlegement	Alrm...	0	26/02/1999 - 17:45:37	26/02/1999 - 17:45:39	Mixer maximum level reached: 25 litres	
No Acknowlegement	Alrm...	0	26/02/1999 - 17:45:59	26/02/1999 - 17:46:56	Mixer maximum level reached: 25 liters	
Acknowleged	Ev_dia	0	26/02/1999 - 17:46:50	26/02/1999 - 17:47:03	Mixing time too short (< 5 s)	16#0000
No Acknowlegement	Alrm...	0	26/02/1999 - 17:46:59	26/02/1999 - 17:52:32	Mixer maximum level reached: 25 liters	
No Acknowlegement	Ev_dia	0	26/02/1999 - 17:47:05		Mixing time too short (< 5 s)	16#0000
Non acknowleged	Alrm...	0	26/02/1999 - 17:52:34		Mixer maximum level reached: 25 liters	

ONLINE | RUN | U:SYS | GR7 OK | MODIF | OVR

A diagnostics Viewer is also available with the operations terminal TSX CCX17 v2.7 and the loaded Web server TSX ETY 410/510•.

Compatibility

DFB Diagnostics can be used with PL7 PRO or PL7 Junior, and are compatible with TSX57/PCX57/PMX57 processors (with software versions equal to or above V3.3).

The diagnostics system is only valid for processors with software versions equal to or above V5.0

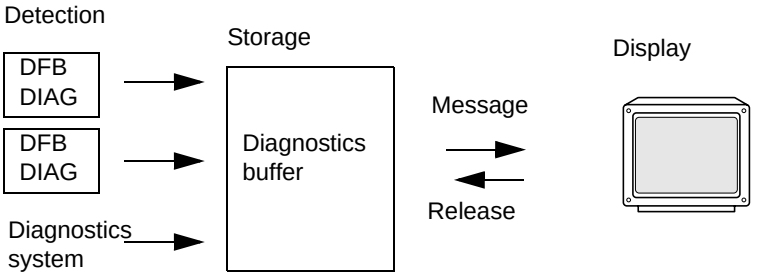
Diagnostics operation in PL7

General

PL7 diagnostics detects errors in monitored elements, and sends error messages to the display systems

Illustration

The following illustration shows the operation of DFB diagnostics:



Operation

The table below describes the different phases of operation:

Phase	Description
1	The DFB diagnostics built into the application program or system detects the conditions surrounding a process error.
2	A diagnostics buffer storage memory called Buffer records errors in the form of messages with the time included.
3	One or several mono-station Viewers (maximum 15) allow: ● message display ● message acknowledgement ● the initialization of animation tables and cross references ● the start-up of associated program editors

Description of diagnostic DFBs

Description

Two types of diagnostic DFBs are offered:

- **Application DFB** which is used to implement procedure monitoring within the application program.
- **Command and Diagnostics System DFB** of operating parts which allow control and command of operating parts elements.

application diagnostics DFB

The application diagnostics DFBs are described in the following table:

Name	Role	Description
EV_DIA	Event monitoring	monitoring the state of 2 bits without taking a time factor into account
MV_DIA	movement monitoring	Monitoring the state of 2 bits without taking into account a time factor, with the possibility of monitoring movement development (change in the state of a bit within a defined time delay).
ALRM_DIA	simplified event monitoring	Interface with diagnostics buffer (error storing)
NEPO_DIA	actuator monitor-command and diagnostics	Monitoring, checking and diagnostics of an operating section element
TEPO_DIA	actuator (linear movement) monitor-command and diagnostics	Monitoring, checking and diagnostics of an operating section element

system diagnostics DFB

The system diagnostics DFB is described in the following table:

Name	Role	Description
ASI_DIA	device monitoring in the AS-i bus	Diagnostics of inputs/outputs AS-i module.
IO_DIA	global monitoring of input/outputs	Diagnostics of all input/output modules

DFB user

PL7 software is also used to create customized DFB diagnostics user function blocks using 2 DFB models.

Descriptive form

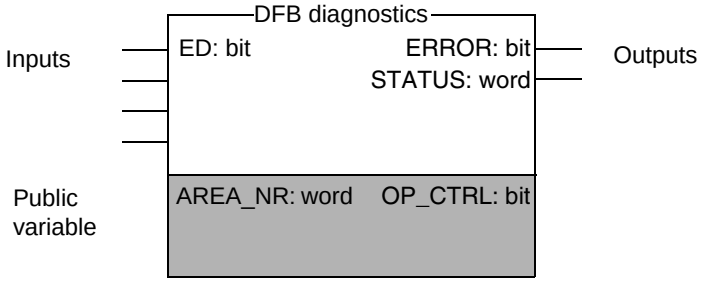
Each diagnostic DFB has a descriptive form showing the DFB function and its parameters (input, output and public variables)

This form can be accessed by double clicking on a DFB type in the application browser, then double clicking on the **Descriptive form** tab in the DFB editor.

General characteristics of DFB diagnostics

DFB Presentation

Diagnostic DFBs are structured as follows:



The quantity of input and output depends on the DFB type: the input/output and the variables described below are common to most of the DFB packages

The descriptive file provides information on the DFB.

Description of input parameters

The following table shows the common input parameter for DFB blocks:

Parameter	Type	Access	Description
ED	bit	R	Enable bit monitor If ED = 0, the DFB inputs are not monitored. ED = 0 by default.

Description of output parameters

The following table shows output parameters common to all DFB blocks:

Parameter	Type	Access	Description
ERROR	bit	R	Fault bit This bit is set at 1 as soon as a fault appears. This bit is set at 0 if the ED input returns to 0 or if there is no longer an error.
STATUS	word	R	Word indicating the type of fault. This word is set at 0 if there is no fault.

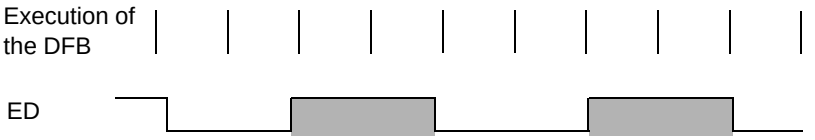
Description of public variables

The following table describes public variables common to DFB blocks:

Parameter	Type	Access	Description
AREA_NR	word	R	This word specifies which. automatic operations are being monitored by the diagnostic DFB. **Example:** • Manufacture: n°1 • Reaming: n°2 • Screwing: n° 3 AREA_NR must possess a value of 1, 2, or 3 to identify which part of the automatic operation has an error. It is advisable to connect the division (see below) to the division in the function module. AREA_NR may take on a value between 0 and 15 (0 is the default)
OP_CTRL	bit	R	This bit signals whether or not DFB instances must be acknowledged by the operator. • OP_CTRL = 0: not acknowledged by the operator, • OP_CTRL = 1: acknowledged by the operator, By default OP_CTRL = 0

Operation

The illustration below shows the diagnostics DFB operating cycle:



Each DFB execution performs the following processes:

- input acquisition (ED,…)
- input monitoring
- output updating (ERROR, STATUS).

How to Program DFB Diagnostics

General

Programming DFB diagnostics requires carrying out a full operation in PL7 Pro or Junior.

Procedure

The following table describes the procedure for programming a DFB .

Step	Action
1	Configure the diagnostics option. **1.** Select the **Station** directory from the application browser. **2.** Access the **Station Properties**dialog box (right-click on **Station** in the application browser and **Properties** in the menu selection). **3.** Select the **Diagnostics** tab, **4.** Tick the box next to **Activate diagnostics in application** . In selecting the diagnostics option, a diagnostics buffer is set up used for storing alarms arising from generated errors or DFB diagnostics. **Note:** If a processor selected is equal to or above version V5.0, choosing the diagnostics option also activates the diagnostics system (see *How to install the diagnostics system, p. 134*)
2	Defining the DFB diagnostics **1.** Import the binary DFB file (UFB file) using the **Import Binary** contextual menu from the DIAG sub-directory, which can be found under the PL7 installation directory (eg *C:\PL7\PL7PRO33\DIAG*), **2.** Define the DFB instance in the PL7 variables editor.
3	Customize the error messages The error message displayed with each DFB diagnostic instance error (except for the `IO_DIA` and `ASI_DIA` DFBs) can be customized. To do this, modify the instance comment defined in the variables editor. **Example**: `Hopper` is a `EV_DIA` DFB instance. The `Hopper` comment is `Empty silo or hopper with open weighing` This is the user error message for the `Hopper` instance. **Note**: Error messages are standard for the `IO_DIA` and `ASI_DIA` DFBs
4	Program the DFB in the contacts network either in LD language or in sequence in ST language. See *DFB programming rules, p. 20*

DFB programming rules

General

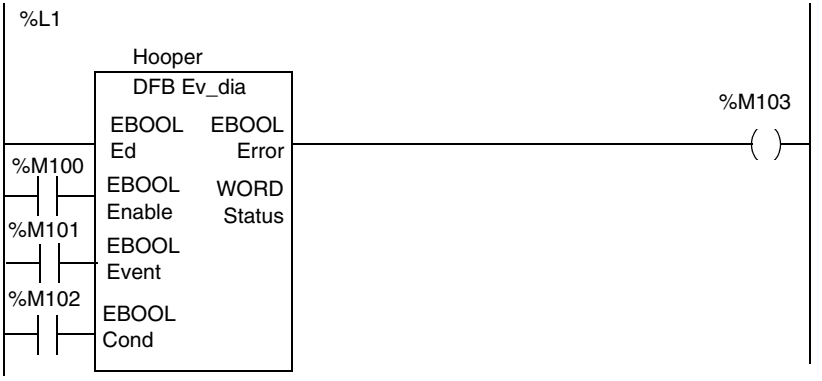
All diagnostic DFBs are capable of self-programming in any program module (Main, SR or section) in language data (LD), structured text (ST) and Instruction List (IL).

Rules

- It is preferable to execute DFB diagnostics in the MAST task (for performance reasons).
- It is strongly recommended to only program a diagnostic DFB instance once within the application.
- To run a diagnostic DFB, it is necessary that:
  - the DFB has been called (the program element to which it is allocated must be executed)
  - ED entry is on 1
- A label is mandatory in the contacts network or in the sequence containing the diagnostic DFB.

Programming in language data

The DFB DIAG function block is placed in a contacts network. The programming connects the input/output



Programming in structured text language

The programming syntax is as follows:

```
%Li: label
Inst ( I1,..., In, O1,...,On );
```

with:

`%Li`: label

`Inst`: name of DFB diag instance

`I1,..., In`: DFB diag inputs

`O1,...,On`: variables connected to DFB diag outputs

Example:

```
! %L1 Hopper (TRUE, Filling, Closed, Level, Klaxon,)
```

with

Inputs

`ED`: always true -> TRUE

`ENABLE`: Filling,

`EVENT`: Closed,

`COND`: Level

Outputs:

`Error`: Klaxon

Programming in Instruction List language

The programming syntax is as follows:

```
[ Inst ( I1,..., In, O1,...,On )]
```

with:

`Inst`: name of DFB diag instance

`I1,..., In`: DFB diag inputs

`O1,...,On`: variables connected to DFB diag outputs

Example:

```
! %L1 LD TRUE
[Hopper (TRUE, Filling, Closed, Level, Klaxon,)]
```

with

Inputs

`ED`: always true -> TRUE

`ENABLE`: Filling,

`EVENT`: Closed,

`COND`: Level

Outputs:

`Error`: Klaxon

How to customize an error message

General

Each DFB has its own standard error message. They can be customized to produce clear and detailed messages which allow easy identification of faults.

Error messages are limited to 40 characters in size.

Standard error messages

The following table shows the standard default message displays: Error messages are displayed on PL7 without accents:

DFB type	Default message display
EV_DIA	"EVENT<>VALUE and/or COND < > 1"
MV_DIA	"EVENT<>VALUE,COND,EVENT_T0,EVENT_T1< > 1"
NEPO_DIA TEPO_DIA	"Configuration error or operating part error"
IO_DIA	"Inputs/outputs fault"
ASI_DIA	standard error message following the faults: "module or bus fault" "At least 1 slave missing" "At least 1 slave does not configure" "At least 1 faulty slave"
ALRM_DIA	"COND1 < > 1 or COND0 < > 0"

Procedure

To set an error message, perform the following:

Step	Action
1	Click on the DFB instances icon in the Variables folder from the browser
2	Select the DFB DIAG instance to be customized
3	Change the comment zone for the DFB instance.

Rules

- Only the first 40 characters are included in the user error message.
- There are no error messages for IO_DIA and ASI_DIA DFBs. There are only standard error messages.
- Viewer displays the user error message if there is one, otherwise it displays the standard error message.
- The standard error message is the same for all DFB instances.
- User error messages can be different for each DFB instance.

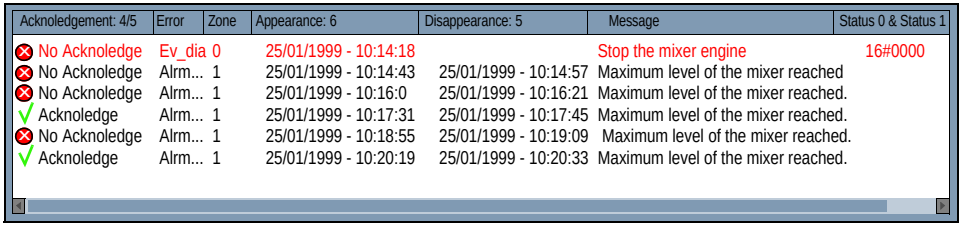
Displaying error messages with the viewer

Introduction

Error messages are displayed by the viewer which is part of PL7.

Illustration

The screen below is an example of the error messages display:



Description of display zones

The following table describes the different display zones:

Zone	Description
Acknowledgement	Message status is shown by an icon and text: ● unacknowledged ● acknowledged, ● without acknowledgement,
Error	States the type of DFB or the system bit (in the case of system diagnostics) which has detected the error
Zone	Indicates the automatic functioning at fault (public variable AREA_NR)
Appearance	Date of the appearance of the error
Disappearance	Date of the disappearance of the error
Message	Displays the error message
Status	Indicates the type of error at the time of the fault detection

The messages appear systematically at the end of the list.

Managing the display

The following elements can be configured (see *How to personalize the viewer messages display, p. 110*):

- The color of the message which has appeared (both text and background color)
- The flashing associated with a message with acknowledgement,
- The choice of zones to be monitored,
- Activation of the archiving function.

Additional error message display functions

Introduction

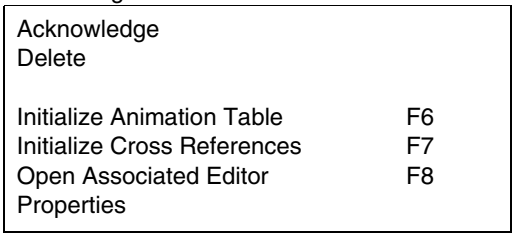
As well as the message display function, the viewer in PL7 provides access via a contextual menu to additional functions which can be used to acknowledge/delete messages, or to identify faults in more detail.

How to access the contextual menu

The contextual menu may be accessed by right-clicking on the error message.

Illustration

The following illustration shows the contextual menu:



Commands description

The following table describes the different commands:

Command	Description
Acknowledge	Acknowledges error messages.
Delete	Deletes the error message from the viewer. **Note:** An error message cannot be deleted if it has not disappeared from the screen or if it has not been acknowledged.
How to initialize an animation table	Displays an animation table which contains input, output and DFB public variables values.
How to initialize cross references	Displays a cross-references table associated with the diagnostics DFB selected.
Open the associated editor	Views the part of the program which contains the diagnostics DFB
Properties	Displays a dialog box which contains detailed information on the detected fault.

Event monitoring: EV_DIA

2

Introduction

Subject of this chapter

This chapter describes the EV_DIA event monitor function block

What's in this Chapter?

This chapter contains the following topics:

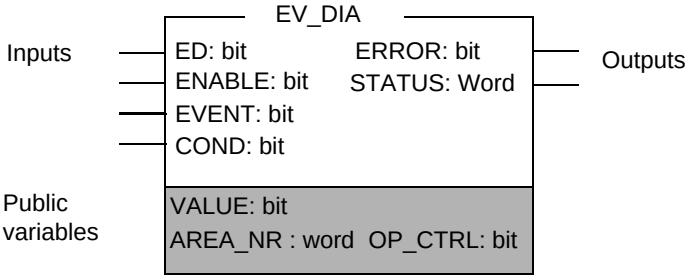
Description of EV_DIA event monitor function block

General

The EV_DIA DFB monitors 2 bit status without

Graphic presentation

This diagram is a graphic representation of the DFB EV_DIA .



Input parameters The following table provides information on the EV_DIA DFB input parameters:

Name	Type	Access by program	Role	Description	Default value
ED	bit	Read	DFB activation bit	If ED = 0, the EVENT and COND input are not monitored._	0
ENABLE	bit	Read	Enable bit monitor	If ENABLE = 0, only COND input is monitored, If ENABLE = 1, the COND and EVENT input are monitored.	0
EVENT	bit	Read	Input bit to be monitored	If the DFB is executed and ENABLE = 1, the DFB verifies that the EVENT input: • possesses the value specified by the VALUE public variable, • is stable (no switching to 1, 0, 1 status one after the other) If this is not the case, the DFB will signal that there is a fault. If ENABLE = 0, theEVENT input is not monitored.	0
COND	bit	Read	Input bit to be monitored	The input bit to be monitored is set at 1, whatever the status of the ENABLE input. If the DFB is executed and this bit changes to 0, the DFB displays a fault.	1

Output parameters The following table shows the output parameters of the EV_DIA DFB:

Name	Type	Access by program	Role	Description
ERROR	bit	Read	Fault bit	This bit is set at 1 as soon as a fault appears. This bit is set at 0 if the ED input returns to 0 or if there is no longer an error.
STATUS	word	Read	Fault type	The following bits indicate the type of fault detected: • bit 0 =1: EVENT is different from the VALUE specified. • bit 1 =1: COND does not possess the expected value • bit 8 =1: EVENT is unstable This word is set at 0 if there is no fault. This word is set at 0 if the ED input returns to 0 or if there is no longer an error.

Public variables The following table provides information on the EV_DIA DFB public variables:

Name	Type	Access by program	Role	Description	Default value
VALUE	bit	Read and Write	Comparison value	Value (0 or 1) to which the EVENT input is compared.	1
AREA_NR	word	Read	Automatic operation zone to be monitored	This word specifies which. automatic operations are being monitored by the DFB diagnostics. Examples: ● Manufacture: n°1 ● Reaming: n°2 ● Screwing n°3 AREA_ NR must have a value of 1, 2 or 3 for the user to identify which part of the automatic process has a fault. It is advisable to connect the division (see below) to the division in the function module. AREA_ NR can take on a value between 0 et 15.	0
OP_CTRL	bit	Read	Acknowledge request	This bit signals whether or not DFB instances must be acknowledged by the operator: ● OP_CTRL = 0: not acknowledged by the operator, ● OP_CTRL = 1: acknowledged by the operator,	0

Function block EV_DIA operation

General functioning

As soon as one of the inputs monitored is no longer parameterized within the DFB, the DFB signals that there is a fault while updating output.

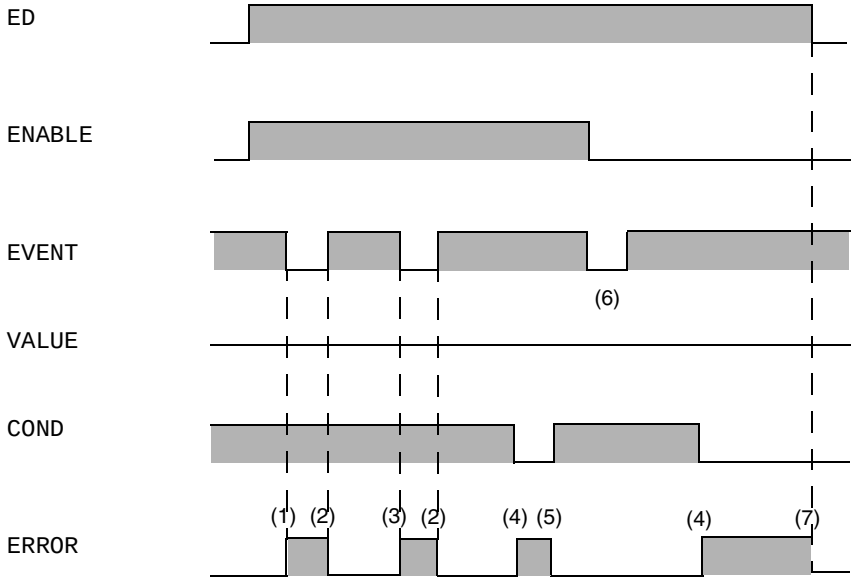
- setting the ERROR bit to 1,
- setting the STATUS word bit to 1 which corresponds with the fault.

Any faults detected during a single monitoring cycle are picked up as they appear (STATUS word bit is set to 1, corresponding with output update).

At the end of a monitoring cycle (falling Edge ED input), the ERROR and STATUS outputs are reinitialized to 0.

Illustration of DFB operation

The following chart shows how the EV_DIA function block works:



Description of operation

The following table describes the different phases illustrated by the chart below:

Phase	Description
1	When the EVENT input is different from the VALUE public variable (ENABLE = 1), a fault is detected.
2	The ERROR output changes to 0 when the EVENT input takes on the values of the VALUE public variable.
3	A fault is detected when the EVENT input becomes unstable. This type of fault appears after the status of EVENT input changes twice in the same monitoring cycle. The EVENT input unstable fault (bit 8 of word STATUS goes to 1) , becomes a EVENT different from VALUE fault (bit 1 of word STATUS goes to 1) if there are more than 1000 PLC cycles before a new fault is detected. The EVENT input unstable fault disappears if there are more than 1000 PLC cycles, and also if the EVENT input is always equal to the value specified by VALUE.
4	A fault is detected when COND input is anything other than 1.
5	ERROR output changes to 0 when COND input takes on the value of 1.
6	EVENT input is different from the VALUE public variable: there is no fault as ENABLE input = 0.
7	ERROR output changes to 0 when ED input takes on the value of 0.

DFB operation during power outage

During a cold restart, the DFB initializes the parameters and public variables:

- COND input set at 1, and other inputs set at 0,
- outputs set at 0,
- VALUE set at 1.

Example of using and programming the EV_DIA function block

Application description

This example describes the monitoring of filling a hopper.

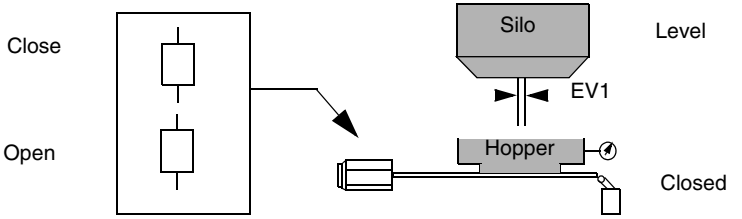
Cycle: pour 100kg of the product into the hopper.

Checks to be carried out

- check that the hopper is closed during filling,
- always check that the silo is not empty.

Illustration of the application

The drawing below shows the application and the checks that have been carried out:



Program PL7

The application is programmed in text, as in this example.

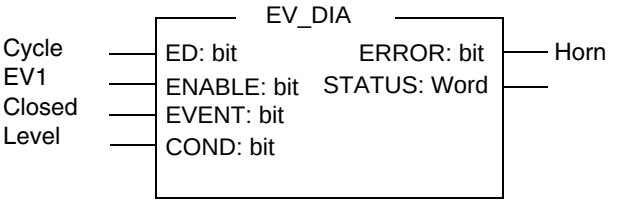
```
%L0:
EV_DIA1 (Cycle, EV1, Closed; Level, Klaxon,);
!IF (Cycle AND Closed)
THEN
 SET EV1;
ELSE
 RESET EV1;
END_IF;
(*Hopper trap door Command *)
!IF Weight >= 100
THEN
 RESET EV1;
 RESET Closure;
 SET Opening;
END_IF;
!IF Weight =0
THEN
 RESET Opening;
 SET Closing;
END_IF;
```

The level in the silo is always checked, as long as the cycle is running.

When the hopper fills EV1 on ENABLE, the hopper trap-door is monitored in its Closed state (EVENT input).

DFB graphic presentation

The illustration below shows a graphic representation of a diagnostics DFB as it is hardwired in this example:



Motion Monitoring: MV_DIA

3

Introduction

Subject of this chapter

This chapter describes the MV_DIA motion monitor function block

What's in this Chapter?

This chapter contains the following topics:

MV_DIA motion monitor function block description

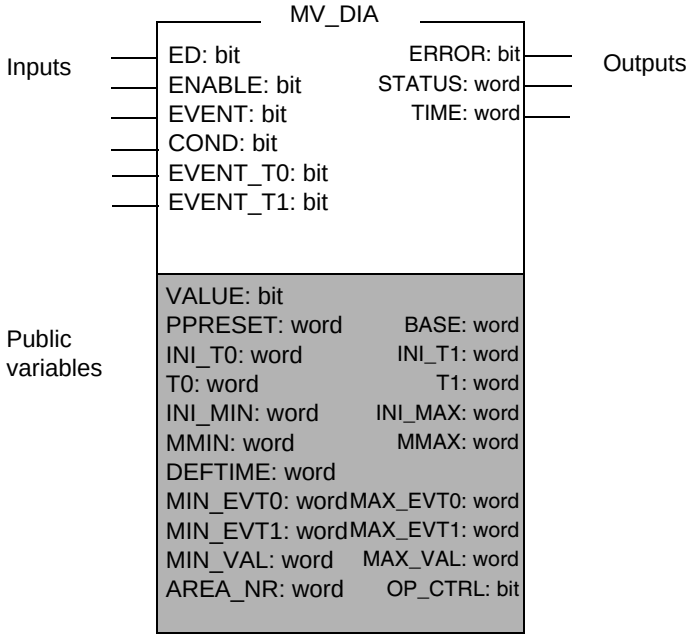
General

The MV_DIA DFB can monitor:

- the state of a bit without time constraints
- a motion (change in bit state within a defined time interval).

Graphic presentation

This diagram is a graphic representation of the DFB MV_DIA:



Description of DFB MV_DIA input parameters

Input parameters The following table describes the MV_DIA DFB input parameters:

Name	Type	Access by program	Role	Description	Default value
ED	bit	Read	DFB activation bit	If ED = 0, the EVENT,EVENT_T0, EVENT_T1 and COND inputs are not monitored.	0
ENABLE	bit	Read	Enable bit monitor	If ENABLE = 0, only COND input is monitored, If ENABLE = 1, the COND, EVENT_T0 and EVENT_T1 inputs are monitored.	0
EVENT	bit	Read	Input bit to be monitored	If the DFB is executed and ENABLE = 1, the DFB verifies that the EVENT input: • possesses the value specified by the VALUE public variable, • is stable (no switching between 1, 0, 1 state). • possesses the value specified by the VALUE public variable, a MMIN minimum time and a MMAX maximum time. If this is not the case, the DFB will signal that there is a fault. If ENABLE = 0, theEVENT input is not monitored.	0
COND	bit	Read	Input bit to be monitored	The input bit to be monitored is set at 1, whatever the status of the ENABLE input. If the DFB is executed and this bit changes to 0, the DFB indicates a fault.	1
EVENT_T0	bit	Read	Exterior event associated with T0 time	This (optional) parameter is a bit which must change from 0 to 1 before T0 time or in ENABLE = 1 format.	1
EVENT_T1	bit	Read	Exterior event associated with T1 time	This (optional) parameter is a bit which must change from 0 to 1 before T1 time or in ENABLE = 1 format.	1

Description of DFB MV_DIA output parameters

Output parameters

The following table shows the output parameters of the MV_DIA DFB:

Name	Type	Access by program	Role	Description
ERROR	bit	Read	Fault bit	This bit is set at 1 as soon as a fault appears. This bit is set at 0 if the ED input returns to 0 or if there is no longer an error.
STATUS	word	Read	Fault type	The following bits indicate the type of fault detected: ● bit 0 =1: EVENT is different from the VALUE specified. ● bit 1 =1: COND does not possess the expected value of 1 ● bit 2 =1: EVENT does not possess the VALUE value during the MIN period requested. ● bit 3 =1: EVENT possesses the value VALUE beyond the MAX time requested ● bit 4 =1: EVENT_T0 not seen at 1 before the T0 time requested ● bit 5 =1: EVENT_T1 not seen at 1 before the T1 time requested ● bit 6 =1: EVENT_T0 not seen at 1 during ENABLE=1 format ● bit 7 =1: EVENT_T1 not seen at 1 during ENABLE=1 format ● bit 8 =1: EVENT is unstable ● bit 9 =1: EVENT_T0 falls back to 0 after T0 time ● bit 10 =1: EVENT_T1 falls back to 0 after T1 time ● bit 14 =1: fault through the overrunning of the internal clock This word is set at 0 if there is no fault. This word is set at 0 if the ED input returns to 0 or if there is no longer an error.
TTIME	word	Read	Current time	Word indicating the current time with a time base expressed in multiples of N x 100 ms. The N coefficient is defined by the BASE public variable. TTIME is initialized at the PPRESET value, and starts to change on the rising edge of the ENABLE input. It stops changing and sets at the value that is running, on the ENABLE falling edge. If a fault is detected (ERROR = 1), TTIME remains frozen in this state until ERROR returns to 0, then: ● if ENABLE = 0, TTIME = 0 ● if ENABLE = 1, TTIME = internal running time

Description of DFB MV_DIA Public Variables

General Public Variables

The table shown hereunder describes the DFB MV_DIA general public variables:

Name	Type	Program access	Role	Description	Default value
VALUE	bit	Reading and writing	Comparison value	Value (0 or 1) compared with the EVENT entry.	1
PPRESET	word	Reading and writing	Initialization value of current time	This word defines the initialization value of the current time (TTIME) on the rising edge of ENABLE by programming or modifying the variable.	0
BASE	word	Reading	Base time value	This word defines the N coefficient required to define the time base. All times are expressed in multiples of N x 100 ms.	1
AREA_NR	word	Reading	Monitored automation zone	This word identifies the automation zone monitored by the DFB tool. Examples: • Fabrication: n°1 • Milling: n°2 • Tapping: n°3 AREA_ NR is required to have the value of 1, 2 or 3 so that the user can identify the default automation section. It is advisable to match the blanking shown above with that in the functional module. AREA_ NR can take a value between 0 and 15.	0
OP_CTRL	bit	Reading	Acknowledgement request	This bit signals whether a DFB acknowledgment is required by the operator: • OP_CTRL = 0: no operator acknowledgment, • OP_CTRL = 1: operator acknowledgment,	0

Public Variables Associated with the EVENT Entry

The table shown hereunder describes the public variables associated with the EVENT entry of DFB MV_DIA:

Name	Type	Program access	Role	Description	Default value
MMIN	word	Reading and writing	Minimum time	This word defines the minimum time during which the EVENT entry must be equal to the internal VALUE data. When the EVENT entry is different from VALUE prior to the MMIN time, the DFB signals a default. If this is the first default in the EVENT entry since the last initialization (ENABLE 0 -> 1), the corresponding time (MMIN) is remembered by DEFTIME.	0
MMAX	word	Reading and writing	Maximum time	This word defines the maximum time during which the EVENT entry must be equal to the internal VALUE data. If the EVENT entry is equal to VALUE after the MMAX time, the DFB signals a default. If this is the first default in the EVENT entry since the last initialization (ENABLE 0 -> 1), the corresponding time (MMIN) is remembered by DEFTIME.	0
DEFTIME	word	Reading and writing	Memorizing the 1st ° default time	This word remembers the time that corresponds with the first default in the EVENT entry. DEFTIME is initialized to 0 as a function of the (0 ou 1) condition of the VALUE variable on the falling/rising edge of the EVENT entry. The DEFTIME variable **cannot** be modified by the program since its default value is **0**.	0
MIN_VAL	word	Reading and writing	Remembering the minimum time	This word memorizes the minimum time for the EVENT entry to have a value specified by the VALUE data. MIN_VAL is set to 32767 on the rising edge of the entry.	32767
MAX_VAL	word	Reading and writing	Remembering the maximum time	This word memorizes the maximum time for the EVENT entry to have a value specified by the VALUE data. MIN_VAL is set to 0 on the rising edge of the ED entry.	0
INI_MIN	word	Reading	MMIN initial value	This word indicates the initial value of the MMIN time. This value is transferred to MMIN for startup or cold boot.	0
INI_MAX	word	Reading	MMAX initial value	This word indicates the initial value of the MMAX time. This value is transferred to MMax for startup or cold boot.	0

Public Variables Associated with EVENT _T0 or T1 Entries

The table shown hereunder describes the public variables associated with the EVENT_Ti entries (with i = 0 or 1) of the MV_DIA DFB:

Name	Type	Program access	Role	Description	Default value
Ti	word	Reading and writing	Minimum time	This word defines the maximum Ti time so that the EVENT_Ti entry passes from state 0 to 1. If this status is changed after Ti time, the DFB signals a default.	0
MIN_EVTi	word	Reading and writing	Remembering the minimum time	This word remembers the minimum time already required for the EVENT_Ti entry to pass from status 0 to status 1. MIN_EVTi is initialized at 32767 on the rising edge of the ED entry.	32767
MAX_EVTi	word	Reading and writing	Remembering the maximum time	This word remembers the maximum time already required for the EVENT_Ti entry to pass from status 0 to status 1. MAX_EVTi is initialized at 0 on the rising edge of the ED entry.	0
INIT_Ti	word	Reading	The initial value Ti time	This word indicates the initial value of Ti time. This value is transferred to the Ti data for startup or cold boot. MIN_VAL is set to 32767 on the rising edge of the entry.	0

Function block EV_DIA operation

General operation

As soon as one of the inputs monitored is no longer parameterized within the DFB, the DFB signals that there is a fault while updating output:

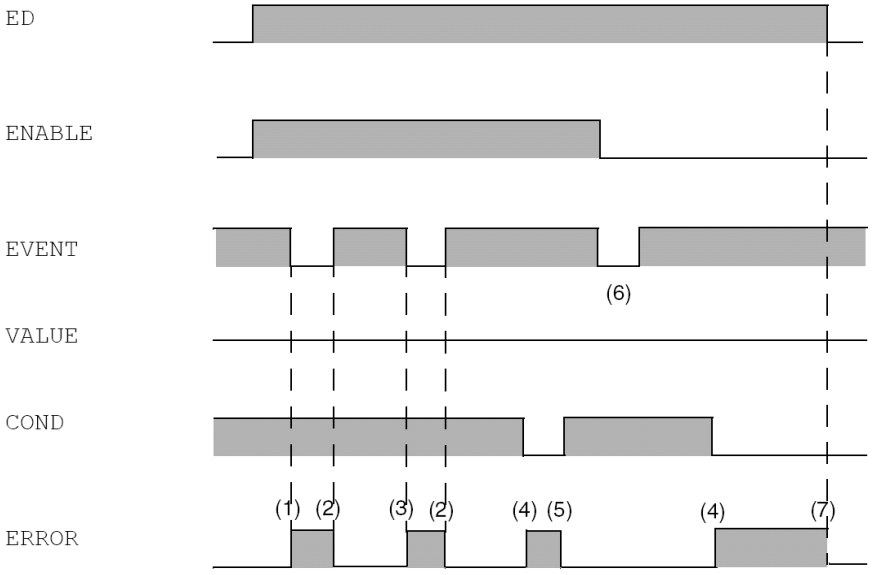
- setting the ERROR bit to 1,
- setting the STATUS word bit to 1 which corresponds with the fault.

Any faults detected during a single monitoring cycle are picked up as they appear (STATUS word bit is set to 1, corresponding with output update).

At the end of a monitoring cycle (falling Edge ED input), the ERROR andSTATUS outputs are reinitialized to 0.

Illustration of DFB operation, EVENT and COND inputs

The following chart shows how the MV_DIA function block works:



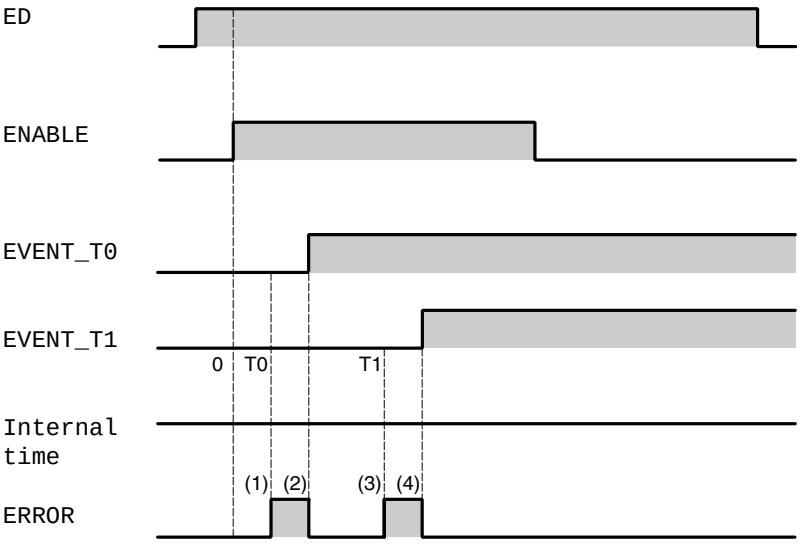
Description of operation, EVENT and COND inputs

The following table describes the different phases illustrated by the chart below:

Phase	Description
1	When the EVENT input is different from the VALUE public variable (ENABLE = 1), a fault is detected.
2	The ERROR output changes to 0 when the EVENT input takes on the values of the VALUE public variable.
3	A fault is detected when the EVENT input becomes unstable. This type of fault appears after the status of EVENT input changes twice in the same monitoring cycle. The EVENT input unstable fault (bit 8 of Status word goes to 1) , becomes an EVENT different from VALUE fault (bit 1 of Status word goes to 1) if there are more than 1000 PLC cycles before a new fault is detected. The EVENT input unstable fault disappears if there are more than 1000 PLC cycles, and also if the EVENT input is always equal to the value specified by VALUE.
4	A fault is detected when COND input is anything other than 1.
5	ERROR output changes to 0 when COND input takes on the value of 1.
6	EVENT input is different from the VALUE public variable: there is no fault as ENABLE input = 0.
7	ERROR output changes to 0 when ED input takes on the value of 0.

Illustration of DFB operation, EVENT_T0 and EVENT_T1 inputs

The following chart shows how the MV_DIA function block works:



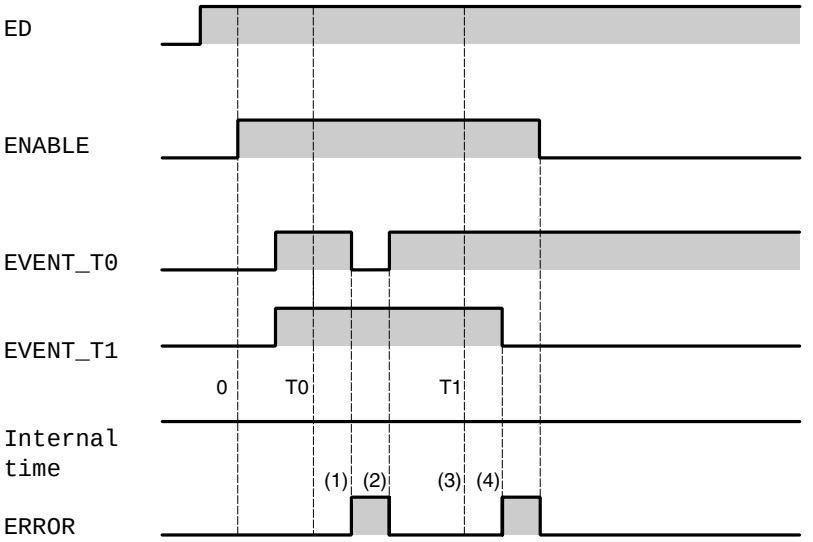
Description of operation for EVENT_T0 and EVENT_T1

The following table describes the different phases illustrated by the chart below:

Phase	Description
1	A fault is detected when EVENT_T0 input does not change to 1 during T0 time.
2	ERROR output changes to 0 when EVENT_T0 input takes on the value of 1.
3	A fault is detected when EVENT_T1 input has not changed to 1 during T1 time.
4	ERROR output changes to 0 when EVENT_T1 input takes on the value of 1.

Illustration of DFB operation, EVENT_T0 and EVENT_T1 inputs

The following chart shows how the MV_DIA function block works:



Description of operation for EVENT_T0 and EVENT_T1

The following table describes the different phases illustrated by the chart below:

Phase	Description
1	A fault is detected when EVENT_T0 input has not remained at 1 after T0 time.
2	ERROR output changes to 0 when EVENT_T0 input takes on the value of 1.
3	A fault is detected when EVENT_T1 input has not remained at 1 after T1 time.
4	ERROR output changes to 0 when ENABLE input changes to 0.

Time base

The time base for counting T0 , T1, MMIN, and MMAX actual time is defined by BASE. A change in the BASE value is not taken into account during the monitoring cycle which is running at the time. It will be taken into account at the start of the next cycle.

DFB operation during power outage

During a cold restart, the DFB initializes the parameters and public variables:

- COND, EVENT_T0 and EVENT_T1 inputs are set at 1
- Other inputs (ENABLE, EVENT) are set at 0
- ERROR, STATUS and TTIME outputs are set at 0
- VALUE is set at 1
- INI_T0, INI_T1, INI_MIN, and INI_MAX are transferred respectively to T0, T1, MMIN and MMAX
- Other data (PPRESET, DEFTIME, MAX_EVT0, MAX_EVT1, and MAX_VAL) are set at 0.

Example of programming and utilization

Application description

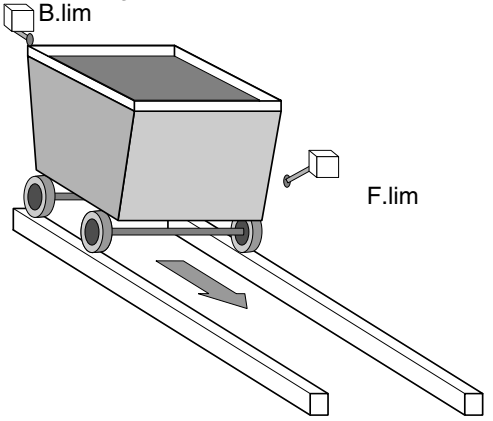
This example describes monitoring the movement of a

Checks to be carried out:

- check that the `Forward` order has been correctly entered,
- after receiving the command `Forward`, ensure that the cart leaves the `fcAr` sensor before 1 second has passed,
- check that the `Forward` run-time period does not go beyond 10 seconds,
- check that the 2 sensors at the end of the run are never on 1 at the same time,
- check that the `fcAr` sensor is on 1 while the cart is at a halt.

Illustration of the application

The drawing below shows the application and the checks that have been carried out:



Program PL7

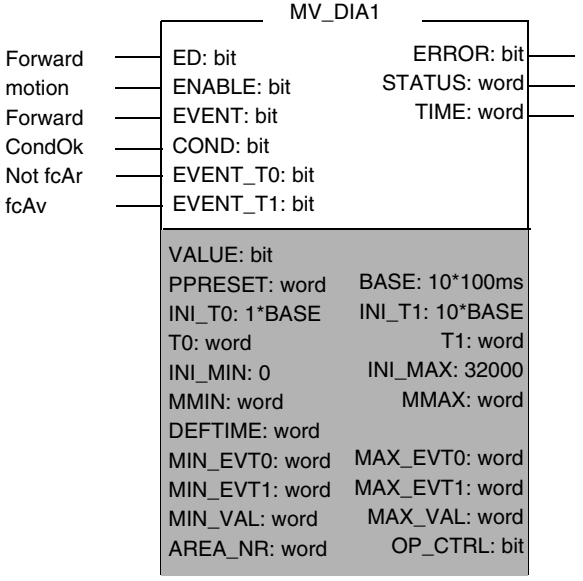
The application is programmed in text, in this example.

```
%L0:
Avance := Avant AND NOT fcAv;
CondOK := Not (fcAv AND fcAr) AND (fcAr OR Avance OR fcAv)
NfcAr := Not fcAr;
MV_DIA1 (Avance, Forward, CondOK, NfcAr, fcAv, , ,) ;
```

- EVENT input is used to check that the Forward order has been correctly entered while the cart moves.
- EVENT_T0 input is used to ensure that the card leaves the fcAr sensor before 1 second,
- EVENT_T1 input checks that the run-time does not last longer than 10 seconds,
- COND input is monitored on 1 for the entire time that the DFB is running. It is used to check that:
  - the fcAr sensor is on 1 while the is at a stop,
  - the two fcAr and fcAv sensors are never on 1 at the same time.

Graphic presentation

The illustration below shows a DFB diagnostics as it is hardwired in this example:



Commands and Diagnostics for the operating part: NEPO_DIA, TEPO_DIA

4

Introduction

Aim of this chapter

This chapter describes the command and diagnostics function blocks NEPO_DIA and TEPO_DIA.

What's in this Chapter?

This chapter contains the following sections:

4.1 Introduction to NEPO_DIA and TEPO_DIA DFBs

Introduction

Aim of this section

This section introduces the command and diagnostics function blocks for the operating part: NEPO_DIA and TEPO_DIA.

What's in this Section?

This section contains the following topics:

Introduction to command and diagnostics function blocks for the operating section: NEPO_DIA and TEPO_DIA

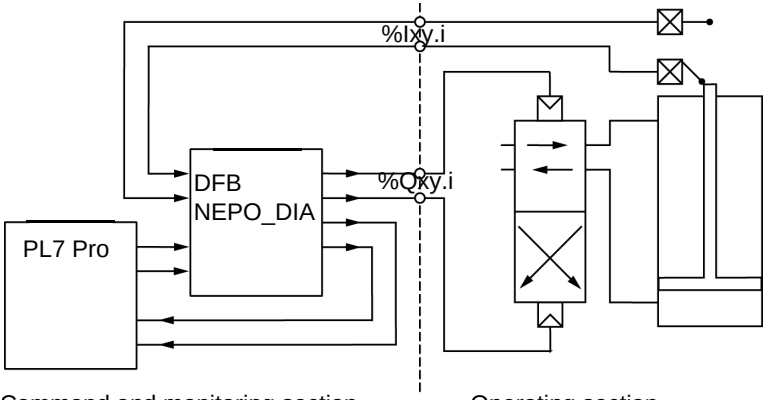
General

These DFBs are used to monitor, command, and carry out diagnostics of an operating part element, i.e. any device which acts directly on manufactured items and the environment.

These DFBs, defined by a "preactuator-actuator/sensor", maintain positioning between two points of reference (whether monitored or not), (for linear or rotating movement) carried out at a constant speed.

Illustration

This drawing illustrates the use of command and diagnostics function blocks in the operating section:



Areas of use

Actual use involves:

- commanding jacks (monostable, bistable or middle point distributers),
- commanding certain positioning motors,
- binding, unit, machining and turntable command.

Particulars of use

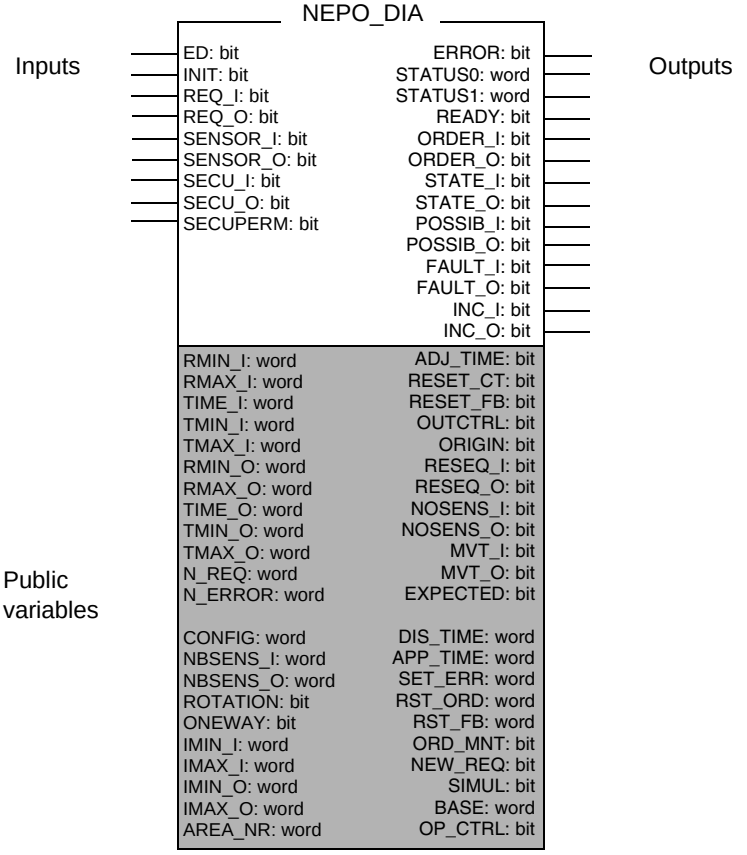
The `TEPO_DIA` DFB is absolutely identical to the `NEPO_DIA` DFB.

Its only limitation is that it can only manage linear movement (i.e. not rotating). As a result, the `ROTATION` and `ONEWAY` public variables do not exist for this DFB.

Description of command and diagnostics function block of operating section: NEPO_DIA and TEPO_DIA

Graphic presentation

This drawing is a graphic representation of the function blocks NEPO_DIA and TEPO_DIA:



Note: The TEPO_DIA DFB does not have ROTATION and ONEWAY public variables.

4.2 Description of NEPO_DIA and TEPO_DIA DFB parameters

Introduction

Aim of this section

This section describes the command and diagnostic function block parameters of the operating section: NEPO_DIA and TEPO_DIA.

What's in this Section?

This section contains the following topics:

Description of NEPO_DIA and TEPO_DIA DFB input parameters

Input parameters The table below describes the input parameters for NEPO_DIA and TEPO_DIA DFBs:

Name	Type	Access by program	Role	Description	Default value
ED	bit	Read	DFB activation bit	When ED = 0, the DFB is not executed.	0
INIT	bit	Read	Fault acknowledgement bit	When on 1, this bit acknowledges the faults indicated by the ERROR bit and the STATUS0 word. It is reset to 0 by the DFB.	0
REQ_I REQ_Q	bit	Read	Request bits	These bits are set on 1 by the command part to request an "input" and "output" movement respectively.	0 0
SENSOR_I SENSOR_O	bit	Read	Information input bit	These inputs receive position information from all the "input" and "output" position sensors respectively.	0
SECU_I SECU_O	bit	Read	Safety condition	These inputs are used to link up the safety conditions of "input" and "output" movements respectively.	0
SECUPERM	bit	Read	Operating condition	This input is used to link up continuous operating conditions.	0

Description of NEPO_DIA and TEPO_DIA DFB output parameters

Output parameters

The table below describes the output parameters for NEPO_DIA and TEPO_DIA DFBs:

Name	Type	Access by program	Role	Description	Default value
ERROR	bit	Read	Fault bit	This bit is set on 1 as soon as a fault appears and providing that the fault has not been masked. See *Selection mask for public variables, p. 68*)	0
STATUS0 STATUS1	word	Read	Fault type	These two words indicate the type of fault. STATUS0 indicates faults linked to DFB operation; STATUS1 is kept for configuration faults. (See *Description of NEPO_DIA and TEPO_DIA DFB status words, p. 58*)	0
READY	bit	Read	DFB availability	• When On 1, The Dfb Is In Command Mode (Setting Orders). • When On 0, The Dfb Is In Recalibration Mode (Awaiting Reference Point).	0
ORDER_I ORDER_O	bit	Read	Activation indicator	When on 1, these bits indicate that the "input" and "output" commands have been activated respectively.	0
STATE_I STATE_O	bit	Read	Input position	When on 1, these bits indicate that the "input" and "output" commands are being checked.	0
POSSIB_I POSSIB_O	bit	Read	Availability indicator	These bits indicate that the DFB is ready to accept "input" and "output" movement requests respectively.	0
FAULT_I FAULT_O	bit	Read	Fault bit	These bits indicate a constant fault during "input" and "output" movements (inoperative position) respectively.	0
INC_I INC_O	bit	Read	Fault bit	In the absence of an order or request, these bits indicate an incoherence respectively: • between the "input" state awaited by the automatic process (RESEQ_1 or ORIGIN data) and the position recorded by the DFB. • between the "output" state awaited by the automatic process (RESEQ_0 data) and the position recorded by the DFB.	0

Description of NEPO_DIA and TEPO_DIA DFB status words

Introduction

When the DFB detects a fault, the latter is indicated via the words STATUS0 and STATUS1 (several faults can be indicated at the same time).

Whether faults are latched or not depends on the selection mask values of the DFB operation when the fault happens. RST_ORD and RST_FB:

- a fault selected in RST_FB will be recorded in STATUS0 until it disappears and is acknowledged by INIT (the DFB changes to shift mode),
- a fault selected in RST_ORD will be recorded in STATUS0 until it disappears and is acknowledged by INIT (the DFB remains in monitor/command mode),
- all other (nonselected) faults will stop being indicated when the cause of the fault disappears.

A fault selected in SET_ERR sets the ERROR bit to 1.

Status word 0

The following table describes the meaning of status word 0 bits for the NEPO_DIA and TEPO_DIA DFBs.

Bit	Error	Description
bit 0 = 1	Error on command or abnormal sensor information	The DFB has detected an aberrant command or incoherent information about positions. Aberrant commands: "input" and "output" requests present at the same time, using the "input" command for a monostable actuator with a single request, expected "input" (RESEQ_1) and "output" RESEQ_0 states present at the same time. Incoherent position information: nonmerged position sensors for a rotating movement, unchecked position with active position sensor, a position being checked by several sensors, and SENSOR_I/O and NOSENS_I/O variables active at the same time.
bit 1 = 1 bit 2 = 1	Unexpected "input" sensor Unexpected "output" sensor	When in position, at least one opposing position sensor is active during time beyond the authorized timing, configured in APP_TIME. After having reset, the sensor for the position just left reappears during a time period outside the authorized time, which is defined in APP_TIME .During recalibration, at least one sensor is present at each position.
bit 3 = 1 bit 4 = 1	Mistimed "input" sensor Mistimed "output" sensor	At least one go to position sensor is present before the minimum movement time, defined in RMIN_I or RMIN_O .
bit 5 = 1 bit 6 = 1	Late "input" sensor Late "output" sensor	At least one go to position sensor is not yet in place beyond the maximum time given to a movement, and defined in RMAX_I or RMAX_O.
bit 7 = 1 bit 8 = 1	"Input" sensor disappearance "Output" sensor disappearance	While in position, at least one sensor has disappeared during a period outside the tolerated time, configured in DIS_TIME . While recalibrating, no position is found.
bit 9 = 1	Condition of continuous disappearance	Continuous conditions have disappeared during a movement.
bit 10 = 1 bit 11= 1	Disappearance of the safety condition for the "input" movement Disappearance of the safety condition for the "output" movement	The safety condition has disappeared during a movement.

Bit	Error	Description
bit 12 = 1 bit 13= 1	"Input" request refused "Output" request refused"	A request cannot be accepted by the DFB (safety conditions and/or continuously absent conditions etc..)
bit 12 = 1 bit 13= 1	"Input" sensor which has not reset "Output" sensor which has not reset	At least one sensor for the position just left has not reset after the minimum movement time, defined in RMIN_I or RMIN_O .

Status word1

Status word 1 detects configuration faults. During a DFB initialization (i.e. application transfer, cartridge changing etc…), the DFB is in "outside operation mode", and is awaiting the reference point At this point it can detect configuration errors which impede its operation, and they are indicated by the STATUS1 output parameter.

The following table describes the meaning of status word 1 bits for the NEPO_DIA and TEPO_DIA DFBs:

Bit	Description
bit 0 = 1	Invalid actuator type (faulty CONFIG value).
bit 1 = 1	ET "input" position and "output" position selected are not monitored.
bit 2 = 1	ET rotating movement, one of the selected positions is not monitored.
bit 3 = 1	Rotating movement, monostable and in one direction only.
bit 4 = 1	Maximum duration of a movement below or equal to the minimum duration.
bit 5 = 1	Modeling and training modes of movement periods.
bit 6 = 1	Translation and single directional movement only
bit 7 = 1	Training mode for movement periods and nonmonitored positions
bit 8 = 1	Rotating movement and positions monitored differently.
bit 9 = 1	Incompatible selected CONFIG and ET RST_ORD selection mask.
bit 10 = 1	The CONFIG selected and ET unmonitored position are incompatible (actuator types 2, 7 or 11 and NBSENS_I or NBSENS_O = 0).
bit 11 = 1	Selection masks RST_ORD and RST_FB are incompatible. (the errors selected in RST_FB must also be selected in RST_ORD).
bit 12 = 1	Selection masks RST_ORD, RST_FB and SET_ERR are incompatible. (the errors selected in RST_FB and RST_ORD must also be selected in SET_ERR).
bit 13 = 1	ET rotating movement selection mask RST_FB are incompatible. (ROTATION = 1 and the error sensor(s) not reset and not selected in RST_FB).

Description of the NEPO_DIA and TEPO_DIA DFB time management variables

General

The time management public variable values express a time equal to N x 100 ms, where N is the value of the BASE constant.

Permitted values are integer numbers between 0 and 32767 inclusive

Time management public variables

The following table describes the time management public variables:

Name	Type	Access by program	Role	Description	Default value
RMIN_I RMIN_0	word	Read and Write	Minimum duration reference	These 2 words act as the minimum duration reference for "input" and "output" movements respectively. By default or on RESET_FB request, these words are initialized from the values of IMIN_I and IMIN_0 (or at 0 if IMIN_I = IMAX_I = 0, IMIN_0 = IMAX_0 = 0) respectively.	0
RMAX_I RMAX_0	word	Read and Write	Maximum duration reference	These 2 words act as the maximum reference for RMAX_0 and "input" and "output" movements respectively. By default or on RESET_FB request, these words are initialized from the values of IMAX_I and IMAX_0 (or from 32767 if IMIN_I = IMAX_I = 0, IMIN_0 = IMAX_0 = 0) respectively.	0
TIME_I TIME_0	word	Read	Time	These two words contain the current time for the "input" and "output" movements in progress respectively, or the time the last "input" and "output" movements occurred individually.	0
TMIN_I TMIN_0	word	Read	Automatic operation zone to be monitored	These 2 words memorize the minimum amount of time that was required for the "input" and "output" movements respectively. By default or on RESET_CT request, TMIN_I and TMIN_0 take on the RMAX_I or RMAX_0 value if ADJ_TIME = 1; and IMAX_I or IMAX_0 if ADJ_TIME = 0.	0
TMAX_I TMAX_0	word	Read	Acknowledge request	These 2 words memorize the maximum amount of time that was necessary for the "input" and "output" movements respectively. By default or on RESET_CT request, TMAX_I and TMAX_0 take on the RMIN_I or RMIN_0 value if ADJ_TIME = 1; and IMIN_I or IMIN_0 if ADJ_TIME = 0.	0
IMIN_I IMIN_0	word	Read	Minimum time	These 2 words define the minimum authorized time for the "input" and "output" movements respectively. On DFB initialization, the values of IMIN_I and IMIN_0 are recopied into RMIN_I and RMIN_0 respectively (if IMIN_I and IMIN_0 are not all 2 to 0).	0

Name	Type	Access by program	Role	Description	Default value
IMAX_I IMAX_0	word	Read	Maximum time	These 2 words define the maximum authorized time for the "input" and "output" movements respectively. On DFB initialization, the values of IMAX_I and IMAX_0 are recopied into RMAX_I and RMAX_0 respectively (if IMAX_I and IMAX_0 are not all 2 to 0).	0
DIS_TIME	word	Read	Duration of sensor disappearance	This word defines the duration in which the disappearance of a position sensor is permitted.	0
APP_TIME	word	Read	Duration of sensor appearance	This word defines the duration in which the unexpected appearance of a position sensor is permitted.	0
BASE	word	Read	Base time coefficient	This word represents the N coefficient required for defining the time base. All times are expressed in multiples of N x 100 ms.	1

Description of NEPO_DIA and TEPO_DIA DFB specific request variables

Specific request public variables

The following table describes the public variables used for specific requests:

Name	Type	Access by program	Role	Description	Default value
RESET_CT	bit	Read and Write	counter reinitialization	Set on 1, this bit reinitializes the counters while memorizing the minimum, maximum and actual time of "input" and "output" movements (TMIN_I, TMIN_O, TMAX_I, TMAX_O, TIME_I and TIME_O), the number of movement requests accepted (N_REQ), and the number of errors detected (N_ERROR). It is reset to 0 by the DFB.	0
RESET_FB	bit	Read and Write	DFB reinitialization	Set on 1, this bit reinitializes the DFB (except for data managed by RESET_CT). It is reset to 0 by the DFB.	0

Description of NEPO_DIA and TEPO_DIA DFB configuration variables

Configuration public variables for actuator types

The following table describes the public variables used to configure controlled actuator types:

Name	Type	Access by program	Role	Description	Default value
CONFIG	word	Read	Type of actuator configuration	This word is used to configure the type of actuator command (see following table). By default, CONFIG = -1 (this value is deliberately faulty in order to make choosing the actuator type mandatory).	-1
NBSENS_I NBSENS_O	word	Read	Position Monitoring	These 2 words are used to define how the DFB monitors "input" and "output" positions respectively: ● NBSENS_I (or NBSENS_0) = 0; position is not monitored, ● NBSENS_I (or NBSENS_0) = 1; position is monitored with SENSOR_I (or SENSOR_0)input, ● NBSENS_I (or NBSENS_0) = 2; position is monitored with SENSOR_I (or SENSOR_0) input (working state of all sensors) and public variable NOSENS_I (or NOSENS_0) (resting state of all sensors).	1
ROTATION	bit	Read	Movement type	This bit defines a rotating movement when set on 1. This parameter does not exist for the TEPO_DIA DFB	0
ONEWAY	bit	Read	Movement sequence	When on 1, this bit defines a rotating movement with the possibility of linking up several movements in the same direction. This parameter does not exist for the TEPO_DIA DFB	0
SIMUL	bit	Read	Modeling mode	When on 1, this bit sets the DFB into modeling mode.	0

Selecting the actuator type

The CONFIG internal constant value is used to select the actuator and order type. The following different configurations are possible:

CONFIG	Actuator	Command	Command logic
0	monostable actuator, single order (ORDER_0)	single request (REQ_0)	order if requested (type 1)
1	monostable actuator, single order (ORDER_0)	two requests (REQ_0, REQ_I)	order maintained until reverse order (type 2)
2	monostable actuator, single order (ORDER_0)	two requests (REQ_0, REQ_I)	order if request and order clash in position, locking on reverse request or loss of position (type 5)
3	bistable actuator two separate orders (ORDER_0, ORDER_I)	two requests (REQ_0, REQ_I)	order if requested (type 1)
4	bistable actuator two separate orders (ORDER_0, ORDER_I)	two requests (REQ_0, REQ_I)	order maintained until reverse order (type 2)
5	bistable actuator two separate orders (ORDER_0, ORDER_I)	two requests (REQ_0, REQ_I)	order if request and position not achieved (type 3). The pre-actuator reacts on a pulse, pointless to maintain an order
6	bistable actuator two separate orders (ORDER_0, ORDER_I)	two requests (REQ_0, REQ_I)	order maintained until reverse request and in position (type 4)
7	bistable actuator two separate orders (ORDER_0, ORDER_I)	two requests (REQ_0, REQ_I)	order if request and order clash in position, unlocking on reverse request or loss of position (type 5)
8	multistable actuator with two separate orders (ORDER_0, ORDER_I)	two requests (REQ_0, REQ_I)	idem 4
9	multistable actuator	two requests (REQ_0, REQ_I)	idem 6
10	multistable actuator	two requests (REQ_0, REQ_I) and absence of request	idem 5 Intermediate stop permitted (absence of request)
11	multistable actuator	two requests (REQ_0, REQ_I) and absence of request	idem 7 Intermediate stop permitted (absence of request)

Note: CONFIG = 8 to 11: intermediate stop possible through the default selected in RST_ORD.

Description of the NEPO_DIA and TEPO_DIA DFB fault management variables

Fault management public variables

The following table describes the public variables used to configure the DFB action during an error:

Name	Type	Access by program	Role	Description	Default value
SET_ERR	word	Read	Error selection	This word is used to select the errors which set the ERROR bit to 1.	H'0FE7'
RST_ORD	word	Read	Reset orders to 0	Reset orders to zero (ORDER_I et ORDER_O). These errors are stored in STATUS0 until they are acknowledged. They must also be selected in the SET_ERR mask.	H'0F87'
RST_FB	word	Read	Error selection	This word is used to select the faults which put the DFB in recalibration mode. These errors are stored in STATUS0 until they are acknowledged. They must also be selected in the SET_ERR mask.	H'0187'

Selection mask for public variables

The following table gives the selection mask default values for the SET_ERR, RST_0RD and RST_FB variables:

Bit	Meaning	SET_ERR (H'0FE7')	RST_0RD (H'0F87')	RST_FB (H'0187')
0	Command error	X	X	X
1	Unexpected "input" sensor	X	X	X
2	Unexpected "output" sensor	X	X	X
3	Mistimed "input" sensor	-	-	-
4	Mistimed "output" sensor	-	-	-
5	Late "input" sensor	X	-	-
6	Late "output" sensor	X	-	-
7	"Input" sensor disappearance	X	X	X
8	"Output" sensor disappearance	X	X	X
9	Condition of continuous disappearance	X	X	-
10	"Input" safety cond. disappearance	X	X	-
11	"Output" safety cond. disappearance	X	X	-
12	"Input" request refused	-	-	-
13	"Output" request refused"	-	-	-
14	"Input" sensor which has not reset	-	-	-
15	"Output" sensor which has not reset	-	-	-

Note: A bit indicated by a cross means that it is selected and that the corresponding error is not masked.
The DFB is thus used to execute a movement together with an error and whatever the error may be.
For example, if bit 9, selecting the error; "disappearance of continuous operating conditions", is set to 0, the orders can be activated even if this condition disappears.

Description of the DFB NEPO_DIA and TEPO_DIA control variables

Public variables reliability indicators

The following table describes the public variables used as reliability indicators:

Name	Type	Access by program	Role	Description	Default value
N_REQ	word	Read	Storing the number of requests accepted by the DFB	This word takes value 0 when RESET_CT is set to state 1 or in the event of counter overrun (when limit value 32767 is reached). The counter overrun N_REQ sets both it, and also the N_ERROR counter back to zero	0
N_ERROR	word	Read	Storing the number of errors detected by the DFB	This word takes value 0 when RESET_CT is set to state 1 or in the event of counter overrun (when limit value 32767 is reached). The counter overrun N_ERROR sets both it, and counter N_REQ back to zero.	0

Cycle reset public variables

The following table outlines the public variables used for the cycle reset:

Name	Type	Access by program	Role	Description	Default value
OUTCTRL	bit	Read and Write	Authorization to send orders	After a default selected in RST_FB, this data is used to authorize the DFB to send orders without monitoring the sensors, in order to set the operative section in a checked recalibration position. The SECU_I, SECU_O and SECUPERM inputs must be valid.	0
ORIGIN	bit	Read and Write	Awaiting source position	This bit indicates that the automatic operation is awaiting the "source position" state (equivalent to RESEQ_I but priority).	0
RESEQ_I RESEQ_O	bit	Read and Write	Awaiting state	These 2 bits signal that the automatic operation is awaiting either the "re-input" state or "output" state respectively.	0

Position monitoring public variables

The following table describes the public variables used for position monitoring:

Name	Type	Access by program	Role	Description	Default value
NOSENS_I NOSENS_O	bit	Read and Write	Position monitoring	These bits give the reverse position of the sensors twisted on the respective inputs SENSOR_I and SENSOR_O. These bits are only used if the DFB is configured to monitor the positions with the help of this data (constant internal NBSENS_I and/or NBSENS_O = 2).	

State public variables

The following table outlines the public variables used as status indicators:

Name	Type	Access by program	Role	Description	Default value
ADJ_TIME	bit	Read	Reference time acquisition	This bit signals that the reference times of the movements have been acquired (training mode).	0
MVT_I MVT_O	bit	Read	Transient state of a movement	These 2 bits signal the transient state of a MVT_O movement "re-input" or "output" engaged and not completed (not reached determined position).	0
EXPECTED	bit	Read	Awaiting state	This bit signals that the DFB is waiting for an end of movement sensor to appear (the movement has been engaged for more than RMIN_I or RMIN_O or has been interrupted).	0

Operating mode public variables

The following table describes the public variables used for configuring the DFB on cycle resumption:

Name	Type	Access by program	Role	Description	Default value
ORD_MNT	bit	Read	Error selection	If this bit is at state 1, the orders are reactivated following the disappearance of the indication in STATUS0, or following faults which have reset the orders to 0.	0
NEW_REQ	bit	Read	Reset orders to 0	If this bit is at state 1, new requests are required after the fault detection which has put the DFB in recalibration mode (i.e. detection of a fault selected in RST_FB).	1

Description of NEPO_DIA and TEPO_DIA DFB general public variables

General public variables

The following table describes the general public variables:

Name	Type	Access by program	Role	Description	Default value
AREA_NR	word	Read	Automatic operation zone to be monitored	This word is used to specify which automatic operations zone is being monitored by the diagnostics DFB. Examples: ● Manufacturing: n°1 ● Reaming: n°2 ● Screwing n°3 AREA_NR should have a value of 1, 2 or 3 for the user to identify which section of the automatic operation has a fault. It is advisable to make the division below correspond with the division in the function module. AREA_NR can take on a value between 0 and 15.	0
OP_CTRL	bit	Read	Acknowledge request	This bit signals whether or not a DFB instance must be acknowledged by the operator: ● OP_CTRL = 0: not acknowledged by the operator, ● OP_CTRL = 1: acknowledged by the operator.	0

4.3 Preprogramming NEPO_DIA and TEPO_DIA DFBs

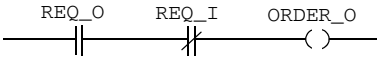
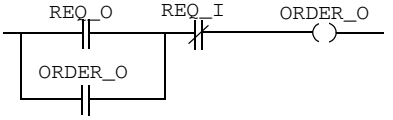
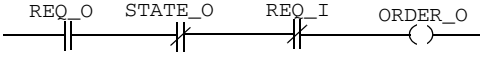
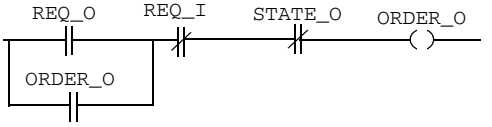
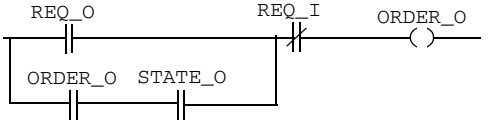
How to preprogram NEPO_DIA and TEPO_DIA DFBs

General

This operation defines `NEPO_DIA` and `TEPO_DIA` DFB functioning.

Procedure

The following table describes the procedure for preprogramming NEPO_DIA ou TEPO_DIA.function blocks:

Step	Actions
1	Select the type of actuator, defined by the CONFIG internal constant: monostable (ORDER_I unused) or bistable (ORDER_O and ORDER_I used),
2	Select the movement type, defined by the ROTATION constant: translation or rotation. If the rotating movement is chosen, the "input" and "output" sensors are merged and the ONEWAY constant defines whether the movement is in one or two direction rotation,
3	Select the type of orders given to the actuator. These orders are applied to the actuators according to the following equations for "output" movements. These equations are the same for "input" movements (replace _O with _I and vice versa): Order if requested (type 1)  Order stored up to reverse request (type 2)  Order if requested and up to position (type 3)  Order stored up to reverse request and position (type 4)  Order if request and order clash on position (type 5) 
4	Select how the physical positions and the element of the operating section are being monitored by the DFB.. It is defined by the NBSENS_O and NBSENS_I internal constants.

Step	Actions
5	Select the action of the DFB on detection of an error: ● SET_ERR data defines the error which sets the ERROR bit to 1, ● RST_ORD data defines the faults which engage ORDER_I et ORDER_0 outputs, ● RST_FB data defines the faults which switch the DFB into recalibration mode. The setting of a bit in either RST_ORD or RST_FB to 1 selects the error associated with the bit of the same rank in STATUS0. ● ORD_MNT data defines whether or not orders should be reactivated following the disappearance of the indication in STATUS0, or following faults which have set orders to 0 during a movement. ● NEW_REQ data defines whether new requests are needed after a fault which has set the DFB to recalibration mode. By default, new requests are demanded.
6	Select the movement duration. ● IMAX_I and IMAX_0 data define the maximum duration of "input" and "output" movements, ● IMIN_I and IMIN_0 data define the minimum duration of "input" and "output" movements, The values express times on a N x 100 ms base, where N is the value of the BASE . On DFB initialization, these values are copied into RMAX_I, RMAX_0, RMIN_I and RMIN_0. If IMIN_I and IMAX_I information (or IMIN_0 and IMAX_0), which define movement duration, are on 0, the DFB will learn this duration.

NBSENS_O and NBSENS internal constants

The table below describes the coding of NBSENS_0 and NBSENS_I internal constants:

NBSENS_0 or NBSENS_I	Monitoring
0	Nonmonitored position This position is considered reached if the DFB is waiting for it to do so, or nonreached if the DFB is not waiting for it to do so. Any fault associated with this position (sensor not reset, not expected) will not be indicated. In other words, this means that a chosen position is not monitored, the DFB will stop the movement (towards this position) as soon as the RMAX_I or RMAX_0 duration limit has been reached, and will examine the EPO potentially. On the other hand, on initialization or recalibration, the reference point can only continue from a monitored position.
1	Position monitored via SENSOR_0 or SENSOR_I input.
2	Position monitored physically with several sensors. The DFB checks the position with 2 information sources SENSOR_0 (or SENSOR_I) and NOSENS_0 (or NOSENS_I), with:POSITION_0 = SENSOR_0 . NOSENS_0 and POSITION_I = SENSOR_I . NOSENS_ISENSOR_0 or SENSOR_I represents the working state of all the sensors. NOSENS_0 or NOSENS_I represents the resting state of all the sensors.

Note: The two positions cannot be chosen, and both cannot be monitored. If this is the case, the DFB indicates that there is a configuration fault (STATUS1) and becomes unusable.

4.4 NEPO_DIA and TEPO_DIA DFB operation

How the command function blocks and the operative section diagnostics work: NEPO_DIA et TEPO_DIA

General

The DFB places itself into the command by maintaining the link between the application program and the action and vice versa:

- Inputs REQ_0 and REQ_I enable requests to be received
- Outputs ORDER_0 and ORDER_I send the orders through to the operator
- Inputs SENSOR_0 and SENSOR_I and if necessary the data NOSENS_0 and NOSENS_I provide the DFB with information on the physical positions of "output" and "re-input".

The movement period is checked through the data RMIN_0, RMAX_0, RMIN_I and RMAX_I .

The inputs SECU_0 and SECU_I set the safety conditions before being accepted during the "re-input" and "output" movements.

Input SECUPERM shows the operating conditions of the machine which have to be accepted during the movements.

Functioning

During normal function (Reset command mode and bit READY = 1), the DFB commands the movement(s) by carrying out the following operations.

Phase	Description
1	sensor check (inputs SENSOR_I and SENSOR_0 and if necessary NOSENS_I and NOSENS_0),
2	Request monitoring (inputsREQ_I and REQ_0)
3	Monitoring of the movement period,
4	Minimum and maximum latch of movement periods,
5	Training on movement periods,
6	Detecting and reacting to errors,
7	Developing the reports for the functional command,
8	Developing the operator commands (outputs ORDER_I and ORDER_0),
9	Updating the functional indicators,
10	Aiding the cycle relaunch.

Movement authorization

Where movement requests are not present and if on the face of it they are authorized (the "request refused" information would not be activated in STATUS0), the DFB positions its outputs POSSIB_I and POSSIB_O in state 1.

Note:
- SECUPERM (continuous operating conditions) or SECU_O/I (movement safety conditions) become part of the bit evaluation POSSIB_O/I if their absence brings the orders down again; or in other words, if the errors included in them are selected in the mask RST_ORD.
- a movement will be refused if an error selected in RST_ORD is present at the time the request is made.
- if the reverse request is present during a movement request, its execution will always be prevented (this error can not be masked). Furthermore, during execution of a movement, a reverse request will cancel the order, whether the request is accepted or not.
- in position a request has no effect on commands of order type up to position (type 3 or 4: POSSIB takes this condition into account.

Information sensor

In position, the disappearance of a sensor is only signaled at the end of the time specified by DIS_TIME. This reset is disabled as soon as a movement request has been accepted.

Outside recalibration mode, the appearance of an unexpected sensor is only signaled after the time specified by APP_TIME .

Information on movement

The DFB positions the data which provide information on the execution of the movement:

- the outputs STATE_I and STATE_0 specify the state of the movement checked by the DFB (position reached). FAULT_I and FAULT_0 signal an error on the movement as it happens
- INC_I and INC_0 signal a discrepancy between the expected position (data RESEQ_I, RESEQ_0 and ORIGIN) and the outputs STATE_I and STATE_0 in the absence of an order or a request,
- the internal data MVT_I and MVT_0 signal that the involved movement has not yet finished (inoperative position).

During movement, the safety conditions linked to the movement and the permanent conditions have to remain enabled according to the masks RST_FB and RST_ORD.

Recalibration mode

Following an error configured in RST_FB or following a request RESET_FB, triggering off a switch to recalibration mode, the DFB performs the operations detailed below:

- deactivation of the bit READY,
- deactivation of outputs STATE_I/O and ORDER_I/O,
- consideration of its configuration data and continuation of the action if there is no configuration error in STATUS1 (only in the event of a request RESET_FB),
- pending request INIT to remove the faults which are no longer present in STATUS0 (only in the event of a fault). The DFB is then in a RESET state whereby it is "Frozen": it no longer tests the constant conditions, the safety conditions and its outputs stop changing,
- switching to recalibration mode to relocate a source position,
- send to reset command mode as soon as it detects a coherent sensor configuration.

Helping to resume the cycle.

The RESEQ_I, RESEQ_O and ORIGIN data inform the DFB of the state expected by the automation.

The DFB stores the last expected state (presented as 1 from RESEQ_I, RESEQ_O or ORIGIN).

If the state or the movement checked by the DFB does not match the expected state (the last one stored), the outputs INC_I and INC_O signal a discrepancy. When the DFB switches into recalibration mode, the states expected before the switch are stored.

Saving the minimum and maximum periods of the movements

For each movement executed, the DFB (in nonsimulated mode) saves the period and stores the minimum and maximum periods in the TMIN_I, TMAX_I, TMIN_O, and TMAX_O data.

The maximum periods are only stored if they are below the maximum reference values RMAX_I and RMAX_O. The RESET_CT data enables the minimum and maximum movement values to be reset.

Training on the movement periods,

It is possible for the DFB to learn the periods of the movements. In order to do this, the time management configuration data must be set at 0.

Whenever a movement is executed without interruption, the data RMIN_0 (or RMIN_I) adopts a value equal to half of the movement period; whilst RMAX_0 (or RMAX_I) adopts a value equal to 1 and a half times this value.

A movement can be said to have been executed without interruption when it has not stopped of its own accord, due either to the absence of requests for operators enabling it, or by default which puts the commands at zero.

Once the periods of the two movements have been established, the ADJ_TIME bit adopts value 1.

Special features of the rotational movement

Position Evaluation

If the two inputs SENSOR_I and SENSOR_0 (and if necessary NOSENS_I and NOSENS_0), are not identical, the "command error" fault will be signaled.

In position, if one or both of the two inputs fall back to 0, the DFB will begin to count the disappearance period of the sensor(s) and will do so until the 2 inputs readopt value 1 in the same time.

During movement, the position will be considered as "quit" if both of the 2 sensors are seen at 0 at least once. The position will be considered as "reached" if both of the sensors are seen at 1.

The only faults signaled concerning the sensor are therefore:

- in position: "disappeared sensor(s)" or "non returning sensor(s)",
- in movement: "misplaced sensor(s)" or "delayed sensor(s)".

Request maintained and position reached

In rotation only one position is checked (the two sensors are taken together). On position and conversely to the translation movement, the two requests are accepted and the two possible movements are begun.

When a movement is finished (position reached), if the "re-input" or "output" request is still present, the movement is automatically restarted. In order to prevent this with the rotation movement, the requests are interpreted on rising edge.

Manual mode

Execution of movements in manual mode (inoperative cycle machine) is dependent on functional command, independent of the DFB. The DFB reacts to the commands in the same way as it does in automatic mode.

However, in order for the DFB to be able to function in manual mode, it has to be executed in exactly the same way outside the machine cycle. In order to do this, if a manual command is anticipated for the DFB, it has to be executed in an easily accessible PL7 module, regardless machine cycle status: module executed during each PLC cycle (POST or SR) whose call can be controlled whilst in function or independently of the machine cycle.

Automatic operation modes

The DFB, during application transfer or cartridge replacement, resets all of its data, takes its configuration data into consideration and goes into recalibration mode (READY at 0).

On %S0 request or restart after a power outage, the DFB goes back into recalibration mode (READY at 0). The outputs ORDER_I/O and STATE_I/O are reset to 0. The counters run by RESET_CT are retained as are the reference times.

The reset-command mode will be activated when a position is found, where no fault is signaled and no request present (whatever the NEW_REQ value).

ASI bus monitoring: ASI_DIA

5

Introduction

Subject of this chapter

This chapter provides a description of the ASI bus monitoring function block: ASI_DIA.

What's in this Chapter?

This chapter contains the following topics:

Description of the function blocks for ASI bus monitoring: ASI_DIA

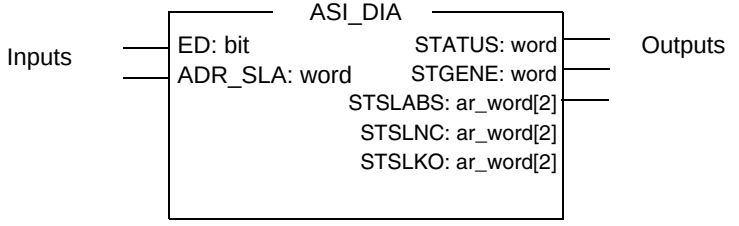
General

This DFB enables error occurrence on the ASI bus to be monitored:

- module or bus default,
- missing slave(s),
- not configured slave(s),
- slave(s) on default.

Graphic presentation

This drawing is the graphic presentation of the function block ASI_DIA:



ar_word[2]: table of 2 16 bit words

Note: ASI faults are saved in zone 0.

Input parameters

The following table shows the input parameters of the ASI_DIA DFB:

Name	Type	Access by program	Role	Description	Default value
ED	bit	Reading	DFB activation bit	If ED = 0, the ASI bus is not monitored.	0
ADR_SLA	word	Reading	Address of the ASI module	This word defines the XY address coding of the ASI module, with: • X = Rack • Y = Module	-

Output parameters

The following table shows the output parameters of the ASI_DIA DFB:

Name	Type	Access by program	Role	Description	Default value
STATUS	word	Reading	Fault type	The next bits indicate the type of fault detected: ● bit 0 = 1: fault module or bus ● bit 1 = 1: missing slave(s) ● bit 2 = 1: not configured slave(s) ● bit 3 = 1: slave(s) on default.	0
STGENE	word	Reading	Current time	Detail of the fault module or bus: ● bit 0 = 1: the ASI module does not give OK response to module identification request ● bit 1 = 1: slave with an address of 0 detected on the bus ● bit 2 = 1: OFFLINE phase active ● bit 3 = 1: DATA_EXCHANGE mode inactive ● bit 3 = 1: no slave presence on the bus.	0
STSLAB[]	ar_word [2]	Reading	List of absent slaves	STSLABS[0]: slaves 0 to 15 ● bit 0: Not significant, still at 0 ● bit 1 = 1: the slave configured to address 1 is absent ● bit 2 = 1: the slave configured to address 2 is absent ● ... ● bit 15 = 1: the slave configured to address 15 is absent. STSLABS[1]: slave 16 to 31 ● bit 0: the slave configured to address 16 is absent ● bit 1 = 1: the slave configured to address 17 is absent ● bit 2 = 1: the slave configured to address 18 is absent ● ... ● bit 15 = 1: the slave configured to address 31 is absent.	0

Name	Type	Access by program	Role	Description	Default value
STSLNC[]	ar_word [2]	Reading	List of not configured slaves	STSLNC[0]: slaves 0 to 15 ● bit 0: Not significant, still at 0 ● bit 1 = 1: the slave detected at address 1 is not configured ● bit 2 = 1: the slave detected at address 2 is not configured ● ... ● bit 15 = 1: the slave detected at address 15 is not configured SSTSLNC[1]: slave 16 to 31 ● bit 0: the slave detected at address 16 is not configured ● bit 1 = 1: the slave detected at address 17 is not configured ● bit 2 = 1: the slave detected at address 18 is not configured ● ... ● bit 15 = 1: the slave detected at address 31 is not configured.	0
STSLKO[]	ar_word [2]	Reading	List of faulty slaves	STSLKO[0]: slaves 0 to 15 ● bit 0: Not significant, still at 0 ● bit 1 = 1: the slave at address 1 is either incorrectly configured or faulty ● bit 2 = 1: the slave at address 2 is either incorrectly configured or faulty ● ... ● bit 15 = 1: the slave at address 15 is either incorrectly configured or faulty SSTSLKO[1]: slave 16 to 31 ● bit 0: the slave at address 16 is either incorrectly configured or faulty ● bit 1 = 1: the slave at address 17 is either incorrectly configured or faulty ● bit 2 = 1: the slave at address 18 is either incorrectly configured or faulty ● ... ● bit 15 = 1: the slave at address 31 is either incorrectly configured or faulty	0

How the ASI_DIA function block works

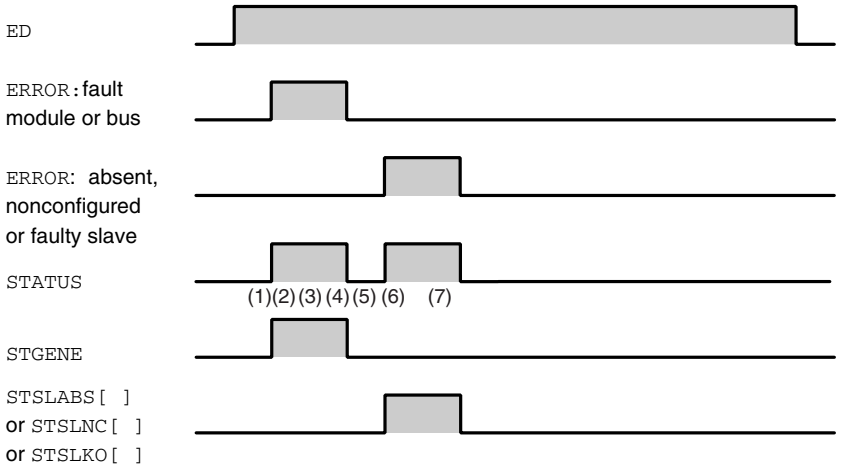
General functioning

All information used in the ASI_DIA DFB is obtained from the language objects associated with the ASI module.

The reading of these language objects intervenes every second in order not to slow down the application's execution.

Illustration of how the ASI_DIA DFB works

The following chart shows how the ASI_DIA function block works:



Description of functions

The following table describes the different phases illustrated by the chart:

Phase	Description
1	A module or bus fault error is saved by the DFB in case of a break in the ASI supply, bit 0 from STATUS and bit 2 of STGENE are set at 1
2	A slave with a 0 address is detected on the bus, bit STGENE is set at 1.
3	Supply is restored to the ASI but the "Fault module or bus" error is not erased because a slave with a 0 address is still detected on the bus.
4	The slave with a 0 address is no longer detected on the bus, the error has disappeared. The STATUS and STGENE special functions are set at 0.
5	An "absent slave(s)", "not configured" or "faulty" error is built into the word STATUS (bit = 1, 2, or 3) and bit 10 of STSLABS[0], STSLNC[0] or STSLKO[0] is built in, indicating which ASI address 10 slave is absent.
6	ASI slave from address 14 is disconnected, only bit 14 of STSLABS[0], STSLNC[0], or STSLKO[0] is set at 1.
7	ASI slaves of address 10 and 14 are again present on the ASI Bus. Bit 1 of STATUS is set at 0 and STSLABS[0], STSLNC[0], or STSLKO[0] is at 0.

AS-i V2 bus monitoring: A2SI_DIA

6

Introduction

Aim of this chapter

This chapter provides a description of the AS-i V2 bus monitoring function block: A2SI_DIA.

What's in this Chapter?

This chapter contains the following topics:

Description of function blocks for AS-i V2 bus monitoring: A2SI_DIA

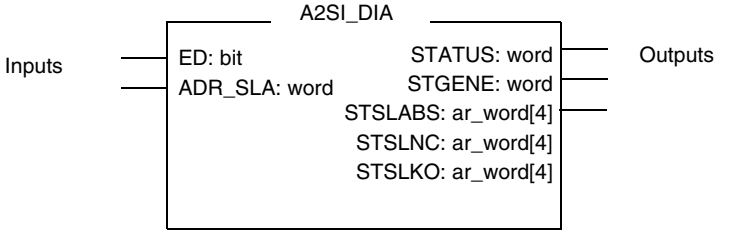
General

This DFB enables error occurrence on the AS-i V2 bus to be monitored:

- module or bus fault,
- missing slave(s),
- non-configured slave(s),
- faulty slave(s).

Graphic presentation

This drawing is the graphic presentation of the function block A2SI_DIA.



ar_word[4]: table of 4 16 bit words

Note: AS-i faults are saved in zone 0.

Input parameters

The following table shows the input parameters of the A2SI_DIA DFB:

Name	Type	Access by program	Role	Description	Default value
ED	bit	Read	DFB activation bit	If ED = 0, the AS-i V2 bus is not monitored.	0
ADR_SLA	word	Read	Address of the ASI module	This word defines the XY address coding of the AS-i V2 TSX SAY 1000 module, with: • X = Rack • Y = Module	-

Output parameters

The following table shows the output parameters of the A2SI_DIA DFB:

Name	Type	Access by program	Role	Description
STATUS	word	Read	Fault type	The next bits indicate the type of fault detected: ● bit 0 = 1: module or bus fault ● bit 1 = 1: missing slave(s) ● bit 2 = 1: non-configured slave(s) ● bit 3 = 1: faulty slave(s). Default value = 0
STGENE	word	Read	Current time	Detail of fault module or bus: ● bit 0 = 1: the ASI module does not give OK response to module identification request ● bit 1 = 1: slave with an address of 0 detected on the bus ● bit 2 = 1: OFFLINE phase active ● bit 3 = 1: DATA_EXCHANGE mode inactive ● bit 3 = 1: no slave presence on the bus. Default value = 0

Name	Type	Access by program	Role	Description
STSLABS[]	ar_word [4]	Read	List of absent slaves	STSLABS[0]: slaves 0 to 15 ● bit 0: not significant, still at 0 ● bit 1 = 1: the slave configured to address 1 is absent ● bit 2 = 1: the slave configured to address 2 is absent ● ... ● bit 15 = 1: the slave configured to address 15 is absent. STSLABS[1]: slaves 16 to 31 ● bit 0: the slave configured to address 16 is absent ● bit 1 = 1: the slave configured to address 17 is absent ● bit 2 = 1: the slave configured to address 18 is absent ● ... ● bit 15 = 1: the slave configured to address 31 is absent. STSLABS[2]: slaves 32 to 47 ● bit 0: the slave configured to address 32 is absent ● bit 1 = 1: the slave configured to address 33 is absent ● bit 2 = 1: the slave configured to address 34 is absent ● ... ● bit 15 = 1: the slave configured to address 47 is absent. STSLABS[3]: slaves 48 to 63 ● bit 0: the slave configured to address 48 is absent ● bit 1 = 1: the slave configured to address 49 is absent ● bit 2 = 1: the slave configured to address 50 is absent ● ... ● bit 15 = 1: the slave configured to address 63 is absent. Default value = 0

Name	Type	Access by program	Role	Description
STSLNC[]	ar_word [4]	Read	List of nonconfigured slaves	STSLNC[0]: slaves 0 to 15 ● bit 0: not significant, still at 0 ● bit 1 = 1: the slave detected at address 1 is not configured ● bit 2 = 1: the slave detected at address 2 is not configured ● ... ● bit 15 = 1: the slave detected at address 15 is not configured. SSTSLNC[1]: slaves 16 to 31 ● bit 0: the slave detected at address 16 is not configured ● bit 1 = 1: the slave detected at address 17 is not configured ● bit 2 = 1: the slave detected at address 18 is not configured ● ... ● bit 15 = 1: the slave detected at address 31 is not configured. SSTSLNC[2]: slaves 32 to 47 ● bit 0: the slave detected at address 32 is not configured ● bit 1 = 1: the slave detected at address 33 is not configured ● bit 2 = 1: the slave detected at address 34 is not configured ● ... ● bit 15 = 1: the slave detected at address 47 is not configured. SSTSLNC[3]: slaves 48 to 63 ● bit 0: the slave detected at address 48 is not configured ● bit 1 = 1: the slave detected at address 49 is not configured ● bit 2 = 1: the slave detected at address 50 is not configured ● ... ● bit 15 = 1: the slave detected at address 63 is not configured. Default value = 0

Name	Type	Access by program	Role	Description
STSLKO[]	ar_word [4]	Read	List of faulty slaves	STSLKO[0]: slaves 0 to 15 ● bit 0: not significant, still at 0 ● bit 1 = 1: the slave at address 1 is either incorrectly configured or faulty ● bit 2 = 1: the slave at address 2 is either incorrectly configured or faulty ● ... ● bit 15 = 1: the slave at address 15 is either incorrectly configured or faulty. SSTSLKO[1]: slaves 16 to 31 ● bit 0: the slave at address 16 is either incorrectly configured or faulty ● bit 1 = 1: the slave at address 17 is either incorrectly configured or faulty ● bit 2 = 1: the slave at address 18 is either incorrectly configured or faulty ● ... ● bit 15 = 1: the slave at address 31 is either incorrectly configured or faulty. SSTSLKO[2]: slaves 32 to 47 ● bit 0: the slave at address 32 is either incorrectly configured or faulty ● bit 1 = 1: the slave at address 33 is either incorrectly configured or faulty ● bit 2 = 1: the slave at address 34 is either incorrectly configured or faulty ● ... ● bit 15 = 1: the slave at address 47 is either incorrectly configured or faulty. SSTSLKO[3]: slaves 48 to 63 ● bit 0: the slave at address 48 is either incorrectly configured or faulty ● bit 1 = 1: the slave at address 49 is either incorrectly configured or faulty ● bit 2 = 1: the slave at address 50 is either incorrectly configured or faulty ● ... ● bit 15 = 1: the slave at address 63 is either incorrectly configured or faulty. Default value = 0

Function block A2SI_DIA operation

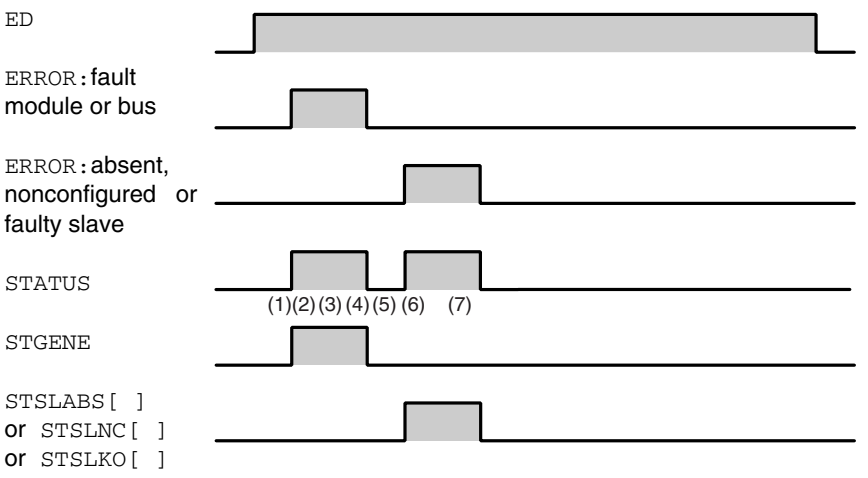
General functioning

All information used in the A2SI_DIADFB is obtained from language objects associated with the **TSX SAY 1000** AS-i V2 module.

These language objects are read every second in order not to slow down the application's execution.

Illustration of how the A2SI_DIA DFB works

The following trend diagram shows how the A2SI_DIA function block works:



Description of operation

The following table shows the different phases illustrated by the chart:

Phase	Description
1	A module or bus fault error is saved by the DFB in case of a break in the AS-i supply, bit 0 from STATUS and the STGENE bit 2 are set at 1
2	A slave with a 0 address is detected on the bus, bit STGENE is set at 1.
3	Supply is restored to the AS-i but the "Fault module or bus" error is not erased because a slave with a 0 address is still detected on the bus.
4	The slave with a 0 address is no longer detected on the bus, the error has disappeared. The STATUS and STGENE special functions are set at 0.
5	An "absent slave(s)", "not configured" or "faulty" error is built into the STATUS word (bit = 1, 2 or 3) and bit 10 of STSLABS[0], STSLNC[0] or STSLKO[0] is built in, indicating which AS-i address 10 slave is absent.
6	AS-i slave from address 14 is disconnected, only bit 14 of STSLABS[0], STSLNC[0] or STSLKO[0] is set at 1.
7	AS-i address slaves 10 and 14 are again present on the AS-i Bus. STATUS bit 1 is set at 0, and STSLABS[0], STSLNC[0] or STSLKO[0] is at 0.

input/output monitoring: IO_DIA

7

Introduction

Subject of this chapter

This chapter provides a description of the IO_DIA input/output monitoring function block.

What's in this Chapter?

This chapter contains the following topics:

Description of the input/output monitoring function blocks: IO_DIA

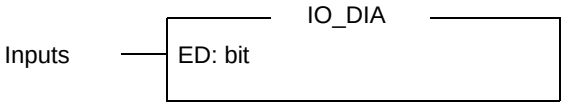
General

This DFB enables the state of the inputs and outputs to be monitored (based on the %S10 bit value).

The `input/output error` message is displayed by the viewer. There is no operator acknowledgement.

Graphic presentation

This drawing is the graphic presentation of the function block `IO_DIA`:



Input parameter

The following table shows the input parameter of the `IO_DIA` DFB:

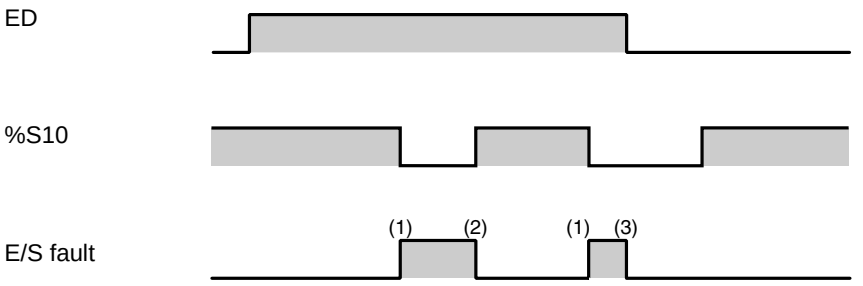
Name	Type	Access by program	Role	Description	Default value
ED	bit	Reading	DFB activation bit	If ED = 1, the %S10 bit (input/output error) is monitored.	0

Note: Input/output errors are saved in zone 0.

How the IO_DIA function block works

Illustration of how the IO_DIA DFB works

The following chart shows how the IO_DIA function block works:



Functioning description

The following table describes the different phases illustrated by the chart:

Phase	Description
1	An input/output error is detected when the %S10 bit system is set at 0
2	Error goes back to zero when the %S10 bit system is set at 1
3	Error returns to zero when the ED input passes 0.

Interface with the diagnostics Buffer: ALRM_DIA

8

Introduction

Aim of this chapter

This chapter provides a description of the ALRM_DIA diagnostics function block.

What's in this Chapter?

This chapter contains the following topics:

Description of the Diagnostics Buffer interface function block: ALRM_DIA

General

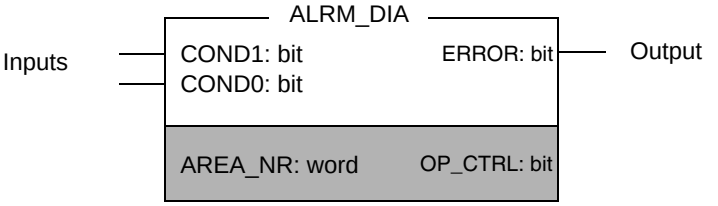
This DFB enables the errors to be stored in a diagnostics buffer:

When COND1 input goes through 0 or when COND0 input goes through 1 it causes the diagnostics buffer to make an error recording.

If neither the COND1 or COND0 input is correct, only one error is recorded. The error disappears when both of the entries COND1 and COND0 return to a correct value.

Graphic presentation

This drawing shows a graphic presentation of the function block ALRM_DIA:



Input parameters

The following table provides information on the ALRM_DIA DFB input parameters:

Name	Type	Access by program	Role	Description	Default value
COND1	bit	Reading	Input bit monitoring at state 1	If the DFB is executed and this bit goes past 0, the DFB displays an error. If the COND0 input goes beyond 1, there is no new error.	1
COND0	bit	Reading	Input bit monitoring at state 0	If the DFB is executed and this bit goes past 1, the DFB displays an error. If the COND1 input goes beyond 0, there is no new error.	0

Output parameters

The following table shows the output parameters of the ALRM_DIA DFB:

Name	Type	Access by program	Role	Description	Default value
ERROR	bit	Reading	Error bit	This bit is set to 1 when an error occurs. This bit is set to 0 if there are no further errors.	0

Public variables

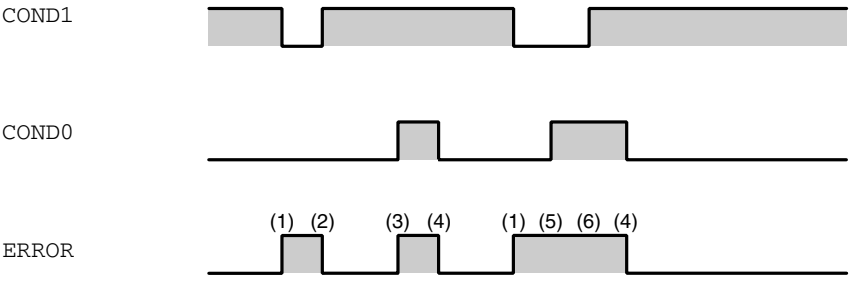
The following table provides information on the ALRM_DIA DFB input parameters:

Parameter	Type	Access	Description
AREA_NR	word	R	This word allows the particular automatic process zone which is being monitored by the diagnostics DFB to be specified. Example: ● Manufacture: n°1 ● Reaming: n°2 ● Screwing: n°3 AREA_NR should have a value of 1, 2 or 3 for the user to identify which part of the automatic process has an error. It is advisable to connect the division shown above to the division in the functional module. AREA_NR can take on a value of between 0 and 15 (0 by default).
OP_CTRL	bit	R	This bit signals whether an acknowlegement of DFB instances needs to be performed by the operator. OP_CTRL = 0: no operator acknowlegement, OP_CTRL = 1: operator acknowledgement, By default OP_CTRL = 0.

How the ALRM_DIA function block works

Illustration of how the ALRM_DIA DFB works

The following chart shows how the ALRM_DIA function block works:



Description of operation

The following table shows the different phases illustrated by the chart:

Phase	Description
1	An error is detected when the COND1 input is set to 0
2	Error is reset to zero when the COND1 input is set to 1.
3	An error is detected when the COND0 input is set to 1.
4	Error is reset to zero when the COND0 input is set to 0.
5	An error is not detected when the COND0 input is set to 1 as there is already an error.
6	The error is not reset to zero when the COND1 input is set to 1 because the COND0 input remains at 1.

Alarm display using the PL7 viewer

9

Introduction

Subject of this chapter

This chapter provides a description of alarm display using the PL7 viewer.

What's in this Chapter?

This chapter contains the following topics:

Introduction to the window message display

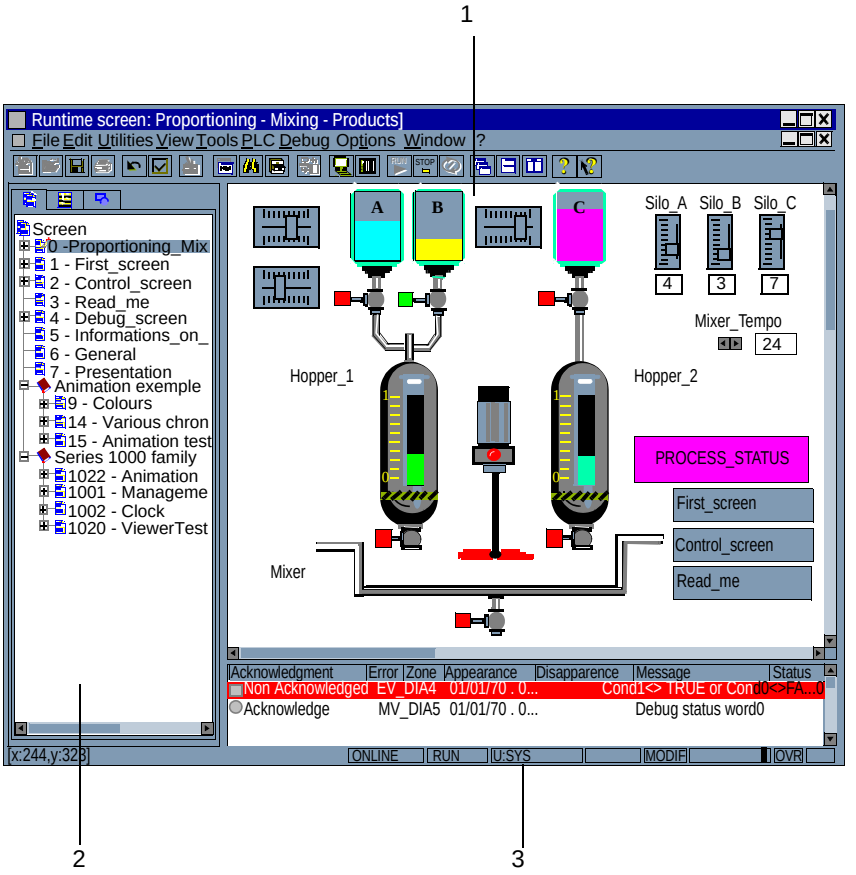
General

The window message display called Viewer appears at the bottom of PL7 operating screens.

Viewer is used in conjunction with debugging an application or an operation. It is used to view faults detected in the application.

Operations screen

The screen below shows the operations screen with the message error window display:



Description

The following table describes the different zones of the operations screen:

Address	Description
1	Graphics editor
2	Screen browser
3	Window Display Viewer It is possible to change the size of this window (by use of the mouse only). However, it is not possible to alter its place on the screen. This window can be hidden. This window is made up of a message list, and may have two scrollbars: ● a vertical scrollbar if the number of messages in the list is more than can be displayed, ● a horizontal scrollbar if the size of Viewer cannot display the whole of a line.

Introduction to Dialog Box Advanced Properties

General

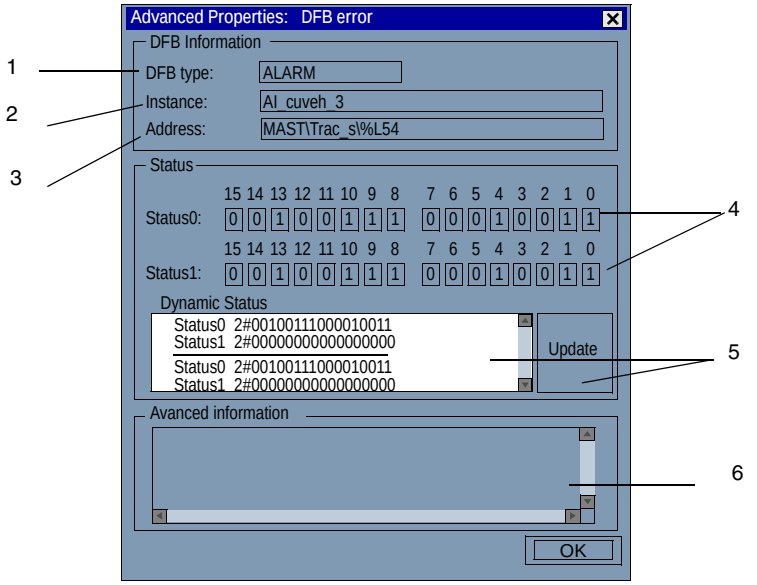
This dialog box provides more detailed information on the fault detected.

To access the dialog box:

- double click on the message
- press the **Enter** key
- or activate the **Properties** command from the contextual menu.

Operating screen

The screen below shows the advanced properties dialog box:



Description

The following table describes the different dialog box areas:

Address	Description
1	Type of DFB error
2	Name of the fault instance
3	Program address where the DFB is entered
4	Content of status words
5	Zone containing status words updated with the **Update** button. The most recent status words displayed appear at the top of the zone (the oldest ones are shifted to the bottom). This information is lost after closing the dialog box.
6	Detailed description of detected faults (text associated with status bits).

How to personalize the viewer messages display

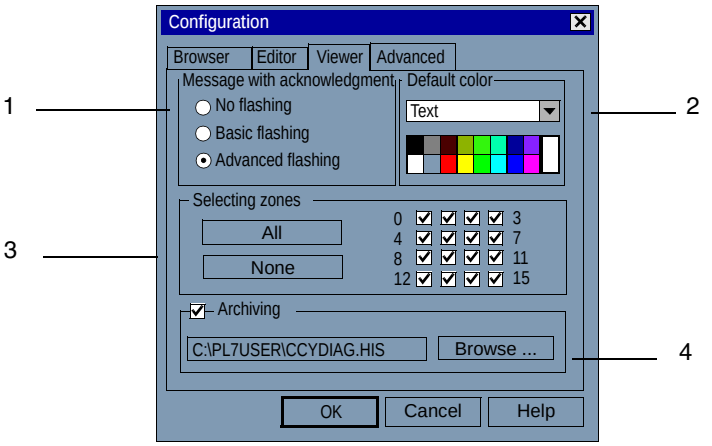
General

The configuration box, which can be accessed by the command **Services** → **Configure** is used for:

- modifying the message color,
- modifying message flashing with acknowledgement,
- selecting the zones to be monitored,
- defining and activating the storing function.

Configuration dialogue box

The screen below shows the message configuration dialogue box:



Description

The following table shows the different functions provided by the configuration dialogue box:

Address	Description
1	Flashes up a message with acknowledgement according to one of the 2 modes: ● simple flashing: only the icon indicating the state of the acknowledgement flashes ● extended line flashing: the entire line flashes. If the entire line flashes, the fill and text colors are simply reversed during flashing.
2	Modifying the text and fill colors. To do this, select **Text** or **Fill** from the dropdown menu, and select a color from the palette.
3	Only displaying messages in the Viewer from a specific or various specific zones. Select the zones to be monitored. The zones are between 0 and 15. All messages (regardless of zone) are displayed in the Viewer by default.
4	Storing messages creates a history file. Select the box labeled Filing to activate this function. It is possible to modify the directory where the history file is located. This file is called NameAppli.his (NameAppli being the name of the actual application) and by default is in the PL7 source (SRC) directory. **How the filing system works:** Messages are filed in line (as soon as a message is read in the PLC buffer, it is written in the file). If a message appears then disappears, it is represented by a single line (message) in the display window, but in the history file it is represented by two. In order to prevent the file from becoming overcrowded, this file is renamed *NameFile.BAK* when it reaches a size of 500 Kb (approx. 5000 saves), and a new history file is created with the original name. If a .BAK file is already in existence, it is replaced without any warning. This file is in ASCII format (each piece of data is separated by a “;”). It is, therefore, easy to import it into any form of text or table processing.

Error Messages Management

General

The viewer is able to:

- sort the message list
- adjust column size
- browse through the list
- acknowledge a message
- deleting a message
- activate another MDI tool.

Message sort

It is possible to sort the message list according to its fields.

To perform a sort, just click on the column heading which contains the data to be sorted. A second click carries out the sort in reverse order (works in similar way to Windows Explorer).

By default, messages are placed in the list in chronological order of the appearance of faults.

Note: Even if the list is sorted according to a given field, a new message appears at the end of the list.

Resizing columns

The list is divided into seven columns, whose sizes can be altered (made bigger or smaller) using the mouse.

If the size of a column is not enough to display all of the information, the text ends with three dots (…).

The size of each column is stored and recalled on opening the Operations Screen tool. The column header blocks also show the number of messages and their status.

Browsing

Browsing within the message list is performed using the keys **Up**, **Down**, **Page Up**, **Page Down**, **Home,** and **End** on the keyboard, or by using the ascender with the mouse if the list contains more messages than can be displayed.

Acknowledgement

To acknowledge a message, just select it and use the corresponding item in the contextual menu with a right click on the mouse.

It is also possible to use the function key **F10** or the button on the services tool bar.

Several messages can be acknowledged simultaneously (with multiple selection). When a message is acknowledged, a command is sent to the PLC and the associated box icon is ticked.

A message can be acknowledged by another Viewer. In this case, the Operating screens tool bar is alerted and the message is displayed as acknowledged.

Deletion of messages in the list.

It is not possible to delete a message which needs acknowledging or a message which has not disappeared.

The **Delete** key or the corresponding item from the contextual menu allows deletion only of messages which have disappeared AND been acknowledged (if necessary).

Activation of another MDI tool

If one (or several) message(s) is selected, it is possible to activate the following MDI tools from the PL7 software workbench:

- animation tables to display external and internal data of a DFB instance (**F6** function key),
- Cross references (**F7** function key),
- Language editor where the DFB instance is referenced by default or by the configuration editor if it concerns a DFB system (**F8** key).

Activating these tools is done either through the contextual menu (right mouse click), via the function keys (**F6**, **F7**, and **F8**) or by using buttons from the **Services** tool bar.

Viewer Operating Mode

Description

The following table describes the viewer's behavior:

Viewer behavior during...	Description
Activation of the Operator screens tool	The display window is initialized (the messages are not saved from one session to the next). If there are messages in the diagnostics buffer during the connection, these will be inserted in the list.
Connection to the PLC	Messages in the list are deleted.
Program transfer to the PLC, or on reconfiguration,	However, if there are messages in the diagnostic buffer, during the connection, these will be inserted in the list.
Disconnection	Messages remain displayed in the Viewer. But messages that must be acknowledged no longer flash and it becomes impossible to acknowledge them.
Memory saturation	The number of messages that can be displayed in the list is only limited by the size of available memory. When the memory becomes insufficient, a message warns the user, and the messages of the faults that have disappeared *and* been acknowledged (if they have to be) are then deleted.

Note: When no viewer explores the buffer diagnostics, all non-present and non acknowledged alarms:

- are deleted from the diagnostics buffer for Premium processors < software version V5.7 and for Micro processors < software version V6.1,
- stay stored in the diagnostics buffer for Premium processors of software version ≥ V5.7 and for Micro processors of software version ≥ V6.1.

Alarm display using the TSX CCX 17 terminal

10

Introduction to the message display window on CCX17

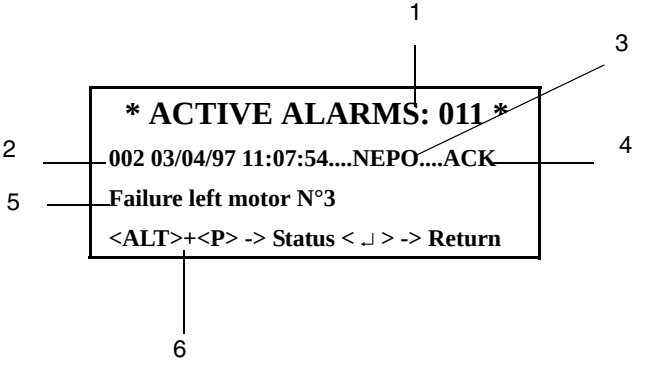
General

The CCX17 terminal is used to view the error messages detected by the diagnostics DFBs.

The alarms are stored in a memory zone with an order number corresponding to their order of arrival. The screen views a group of alarms in real time and when a viewed alarm or an alarm just outside of view disappears, a report is made.

Alarm screen

The screen below shows the error message display screen:



Screen description

The following table describes the different areas of the operating screen:

Address	Description
1	Order number
2	Date and time of fault occurrence
3	DFB type which triggered the alarm
4	Local state of alarm: ● **ACK**: alarm acknowledged (seen by operator). The alarm number is sent back to the PLC, ● **ON**: non acknowledged alarm.
5	Message associated with the alarm.
6	Commands for viewing alarms in the list.

Command description

The following table describes the commands associated with the display screen:

Command	Description
[ALT] + [ACK]	Displays the list of active alarms on the screen with the most recent position.
[¦][Ø]	Moving through the list The checked off alarm is displayed in reverse video.
[ALT] + [¦]	Movement at the end of the list.
[ALT] + [Ø]	Movement at the beginning of the list.
[ACK]	Used to acknowledge the checked off alarm. For a DFB type alarm, if the option has been configured in the DFB instance the acknowledgement data is sent to the PLC.
[ALT] + [P]	If the alarm comes from a diagnostics DFB this is displayed on the status screen. The status information from the DFB which generated the alarm, is displayed on two lines. Use the up and down arrows to look through all the messages. On the final message (final status bit), only the [-] key appears. In the same way that on the first message, only the down arrow appears. In addition to the status messages, the screen displays: the name of the DFB instance and the message associated with the alarm. If the error has disappeared, the message `ALARME DISAPPEARED` is displayed and the status messages are erased.
[ø]	Leaving consultation mode and returning to the actual dialog screen.

Diagnostic user DFB

11

Introduction

Subject of this chapter

This chapter describes how to create and use the diagnostic user function blocks.

What's in this Chapter?

This chapter contains the following topics:

Introduction to the diagnostics user DFB

General

The PL7 software (version > 3.4) can be used to create your own DFB function blocks.

From the 2 DFB models it is possible to create up to:

- 26 process diagnostics user DFBs,
- 26 system diagnostics user DFBs.

Two DFB types

2 types of DFB can be created the:

- **Process**, the code created by the user is command process and application monitoring orientated (for example: to monitor levels with many thresholds).
- **System**, the code created by the user is monitoring module and system orientated (for example: to monitor an axis command module, bits, and system words…).

Display

These DFBs are set without restrictions in the diagnostics supply: they are viewed by the CCX17 Viewer (within the available memory limit) and via the Operating Screens' Viewer. From the selected error message, and using the **Open associated editor F8** command, the Operating Screens Viewer displays:

- the network of contacts or sequence which calls the faulty instance for the process DFBs
- the hardware configuration screen for the system DFB.

Description of the DFB models

General

To create the user diagnostics DFB, the PL7 software provides 2 DFB models:

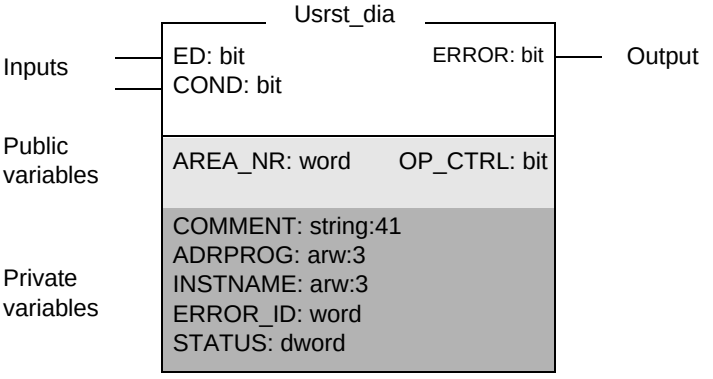
- One written in Contact Language, named *Usrld_dia.ufb*,
- One written in Structured Text language, named *Usrst_dia.ufb*.

These DFBs are commented on (descriptive file), protected (password: diaguser) and sent in binary form.

The interface and code can be enhanced from one of these models to construct the diagnostics DFB suitable for your application.

Graphic presentation

This is a diagram of the model DFB blocks:



Input parameters

The following table shows the input parameters of the DFB model:

Name	Type	Access by program	Role	Description	Default value
ED	bit	Read	DFB activation bit	When ED = 0, the DFB is not executed.	0
COND	bit	Read	Input bit to be monitored	Input bit monitoring at 1. If the DFB is executed and this bit changes to 0, the DFB will signal that there is a fault.	1

Output parameter

The following table shows the output parameters of the DFB model:

Name	Type	Access by program	Role	Description
ERROR	bit	Read	Fault bit	This bit is set to 1 as soon as a fault appears. This bit is set to 0 if the ED input returns to 0 or if there is no longer an error.

General public variables

The following table provides information on the DFB model public variables:

Name	Type	Access by program	Role	Description	Default value
AREA_NR	word	Read	Automatic operation zone to be monitored	This word is used to specify which automatic operations zone is being monitored by the diagnostics DFB. AREA_NR must have a value of 1, 2 or 3 for the user to identify which section of the automatic operation has a fault. It is advisable to make the above division correspond with the division in the function module. AREA_NR can take on a value between 0 and 15.	0
OP_CTRL	bit	Read	Acknowledge request	This bit signals whether or not a DFB instance must be acknowledged by the operator: • OP_CTRL = 0: not acknowledged by the operator, • OP_CTRL = 1: acknowledged by the operator.	0

Private variables The following table provides information on the private variables of the DFB model, the mandatory variables must not be modified:

Name	Type	Access by program	Role	Description	Mandatory variable
COMMENT	string: 41	Read	Error message	The error message displayed by the viewer contains the instance comment entered in the variable editor.	yes
ADRPROG	arw:3	Read	Program address	The program section address calling the faulty DFB instance, this address is used by the viewer to display the contact network or sequence.	yes
INSTNAME	arw:3	Read	Instance name	Name of faulty instance, used to display the faulty instance.	yes
ERROR_ID	word	Read	Identifier no	Identifier no of the error sent via the diagnostics buffer (used to deregister an error).	No
STATUS	dword	Read	Error report	Error Report To Be Controlled By The User.	No

How to create a diagnostics user DFB

Methodology

The following table outlines all the operations necessary to create a diagnostics DFB:

Step	Action
1	Create an application by (by defining a processor of a superior or equal version to V3.3) and configure the diagnostics option (see *How to Program DFB Diagnostics , p. 19*).
2	Import the binary file of a DFB model.
3	Unprotect the DFB and rename the DFB model.
4	Program the DFB type.
5	Get the DFB ready.
6	Debug the DFB. **Note:** A diagnostics DFB requires a label in the language element (contact, network, or sequence) containing its call.
7	Protect the DFB (optional) through " know how " or " modification ": password can be freely chosen.
8	Export the DFB.

Importing the binary file

The following table outlines how to import the (*Usrld_dia.ufb* or *Usrst_dia.ufb*) model diagnostics DFBs:

Step	Action
1	With the aid of the contextual menu (right click on the DFB Types directory in the browser) at the **Import binary** command
2	Select the model file in the DIAG subdirectory which is located in the PL7 installation directory (e.g. *C:\PL7\PL7PRO33\DIAG*)
3	Click on **Import**

Removing protection and renaming the DFBs

The following table details how to remove protection from and rename the (*Usrld_dia.ufb* or *Usrst_dia.ufb*) model diagnostics DFBs:

Step	Action
1	Double click on the DFB model imported in the browser
2	Select the **Not protected** key, and enter the password: diaguser, then validate
3	Validate the DFB
4	Rename the DFB model in the browser: Select the DFB model, then click on the name with the left mouse button and rename this DFB: • `Usr ?_dia` for a process DFB • `Sys ?_dia` for a system DFB. ? being a letter of the alphabet (for example: `Usrf_dia`, `Sysb_dia`).

Binary export of the DFB

The following table details how to remove protection from and rename the diagnostics DFBs created:

Step	Action
1	Access the **Binary export** command using the contextual menu (right click on the DFB in the browser).
2	Select the DIAG subdirectory which is located in the PL7 installation directory (e.g. *C:\PL7\PL7PRO33\DIAG*).
3	Name the DFB: *nom_du_type_DFB.ufb* (ex: *Usra_dia.ufb*) and save.
4	The DFB can then be put back into any user application using binary import.

How to program a diagnostics user DFB type

Steps to follow to program a DFB type

The following table describes the procedure for DFB type programming:

Step	Action
1	Personalize the DFB interface according to the DFB function to be developed.
2	Adapt the DFB model code according to the DFB to be developed.
3	Validate the DFB type.

Rules for personalizing the interface

Failure to respect the following rules can result in serious malfunctioning in all the diagnostics DFBs:

- The private variables `Comment`, `Instname`, and `Adrprog` are mandatory and neither name nor type may be changed.
- The existence of a (private or public) status variable of DWORD type is also mandatory. This variable may or may not be managed by the DFB (but it must exist) and its name can be freely chosen.
- The existence of at least one (private or public) error variable of WORD type is still mandatory. This variable must be managed by the DFB (recording and nonrecording) and its name can be freely chosen.
- Finally, the `Comment` variable has an initial value which is the error message generated by this type of DFB by default. This message may be modified.

Rules for personalizing the code

Modify the DFB model's code according to the DFB function to be developed and modify the error class value corresponding to the name of the DFB. The error class is specified in the code in the REGDFB instruction parameter:

Process type DFB

Name	Code	Name	Code	Name	Code
Usra_dia	16#004A	Usrj_dia	16#0053	Usrs_dia	16#005C
Usrb_dia	16#004B	Usrk_dia	16#0054	Usrt_dia	16#005D
Usrc_dia	16#004C	Usrl_dia	16#0055	Usru_dia	16#005E
Usrd_dia	16#004D	Usrm_dia	16#0056	Usrv_dia	16#005F
Usre_dia	16#004E	Usrn_dia	16#0057	Usrw_dia	16#0060
Usrf_dia	16#004F	Usro_dia	16#0058	Usrx_dia	16#0061
Usrg_dia	16#0050	Usrp_dia	16#0059	Usry_dia	16#0062
Usrh_dia	16#0051	Usrq_dia	16#005A	Usrz_dia	16#0063
Usri_dia	16#0052	Usrr_dia	16#005B	-	-

System type DFB

Name	Code	Name	Code	Name	Code
Sysa_dia	16#008A	Sysj_dia	16#0093	Syss_dia	16#009C
Sysb_dia	16#008B	Sysk_dia	16#0094	Syst_dia	16#009D
Sysc_dia	16#008C	Sysl_dia	16#0095	Sysu_dia	16#009E
Sysd_dia	16#008D	Sysm_dia	16#0096	Sysv_dia	16#009F
Syse_dia	16#008E	Sysn_dia	16#0097	Sysw_dia	16#00A0
Sysf_dia	16#008F	Syso_dia	16#0098	Sysx_dia	16#00A1
Sysg_dia	16#0090	Sysp_dia	16#0099	Sysy_dia	16#00A2
Sysh_dia	16#0091	Sysq_dia	16#009A	Sysz_dia	16#00A3
Sysi_dia	16#0092	Sysr_dia	16#009B	-	-

Note: The REGDFB instruction can only be used in DFBs from the "Template" library.

Instructions for recording alarms

Role

The REGDFB alarm recording instruction entered in DFB code, records the dating of an alarm in the diagnostics buffer.

Syntax

The recording instruction syntax is as follows:

REGDFB(Area_nr, Class ,Slen, Op_ctrl, Comment, Instname, Adrprog, Status, Error_id, Stat).

Input parameters

The following table outlines the role of each recording instruction input parameter:

Name	Role	Type
Area_nr	Machine zone monitored by the DFB: 0 to 15	WORD
Class	Error class (see tables *How to program a diagnostics user DFB type, p. 126*) ● 16#004A to 16#0063 for the process type DFBs ● 16#008A to 16#00A3 for the system type DFBs.	WORD
Slen	Status length: 0, 2, or 4 Bytes: ● 0 = status not managed ● 2 = status managed on one word ● 4 = status managed on a double word.	WORD
Op_ctrl	1= Requested operator acknowledgement 0= No acknowledgement.	BOOL
Comment	Default error message associated with the DFB instance	STRING
Instname	Name of the faulty instance	AR_W
Adrprog	Program address of the faulty DFB instance	AR_W
Status	DFB status (declared as OUT parameter to switch it by address and not by value, must be updated by the DFB)	DWORD

Output parameters

The following table outlines the role of each recording instruction output parameter.

Name	Role	Type
`Error_id`	Error identifier	WORD

`Stat`	Error recording report
	● if recording succeeds: `Stat` = 0 and Error_id is valid
	● if recording fails: `Error_id` is invalid and
	  ● `Stat` = 1: nonconfigured diagnostics buffer
	  ● `Stat` = 2: full diagnostics buffer.
	The %SW160 system word is reserved for receiving the diagnostics DFB registration result (nonmandatory but advisable usage)
		WORD

Instructions for nonrecording of alarms

Role

The DEREG alarm nonrecording instruction entered in DFB code, dates the error disappearance in the diagnostics buffer.

Note: The alarm remains saved in the diagnostics buffer until the fault has been acknowledged (for faults with acknowledgement) and read by all the viewers.

Syntax

The syntax of the recording instruction is as follows:

Result:=DEREG(Error_id)

Note: The DEREG instruction can only be used in DFBs from the "Template" library.

Input parameters

The following table outlines the role of each of the recording instruction input parameters:

Name	Role	Type
Error_id	Error identifier registered previously	WORD

Function return

The following table outlines the role of each recording instruction output parameter:

Name	Role	Type

Result	Error recording report
	• if nonrecording succeeds, Result = 0
	• if nonrecording fails:
	  • Result = 1: diagnostics buffer not configured
	  • Result = 21: incorrect error identifier
	  • Result = 22: no error recorded with this identifier.
	The %SW161 system word is reserved for receiving the result of nonrecording of diagnostics DFBs (nonmandatory but advisable usage).
		WORD

Diagnostics system

12

Introduction

Aim of this chapter

This chapter provides a description of the diagnostics system.

What's in this Chapter?

This chapter contains the following topics:

Introduction to the diagnostics system

Role

This type of diagnostics is used to detect hardware and application faults or system errors.

It is created entirely automatically (without programming) by the PLC processor at system level.

It is used for:

- fault detection,
- formulating the error messages in real time,
- and putting these messages at the display system's disposal.

Operation

The diagnostics system operates the data stemming from bits and system words, and error bits from the input/output modules.

Its error messages are generated when the state of these bits and words changes.

It provides 2 levels of information:

- standard information (message displayed),
- additional information accessible by dialogue box from the PL7 viewer.

List of detected faults

The following table lists all the faults detected by the diagnostics system:

Fault type	Error message	Object monitored	Message number
Hardware	E/S local fault	%Ixy.mod.err	1
	E/S FIPIO fault	%SW128 to 143	1001
	Time error in real time clock	%S51	2001
	Memory card stack error	%S67	2002
	Backup stack error	%S68	2003
Application	Character string error	%S15	10001
	Arithmetic error	%S18	10003
	Index overflow	%S20	10004
	Saturation of the event treatment	%S39	10005
	JUMP to an undefined label	%SW125	10006
	HALT instruction	%SW125	10006
	Error prevention	%SW125	10006
Diagnostics	Saturation of the diagnostics buffer	%S102	10101
Task	Watchdog overrun	%S11	11001
	Task period overrun	%S19	11002
Grafcet	Grafcet table overrun	%S26	12001
Communication	Global FIPIO fault	%SW146	20001

Note: The complete description of bits and system words is provided in the reference manual.

How to install the diagnostics system

Prerequisite

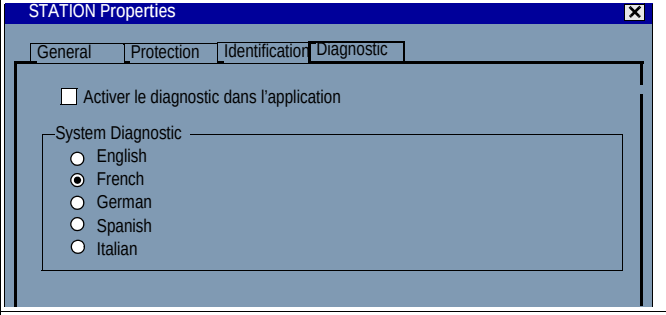
The diagnostics system can only be installed if a version V5 Premium processor has been selected.

The diagnostics system is activated as soon as the diagnostics option is selected.

Note: The diagnostics option activates the diagnostics DFB and diagnostics system use simultaneously.

Procedure

The following table describes the procedure for installing the diagnostics system function:

Step	Action
1	Select the directory **Station** in the application browser.
2	Access the dialogue box **Station Properties** (right click on **Station** in the application browser and **Properties** in the menu selection).
3	Select the **Diagnostics** tab.
4	Tick the box next to **Activate diagnostics in the application**. 
5	Choose the language in which the error messages will be displayed in the viewer and becomes the default PL7 language.

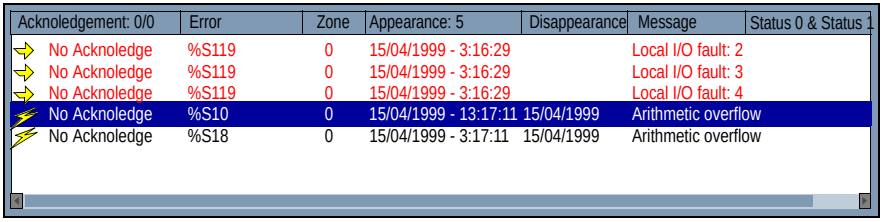
Displaying error messages generated by the diagnostics system

Introduction

Error messages can be displayed by the viewer which is part of PL7 or by other types of viewer (CCX 17...).

Illustration

The screen below is an example of the error messages display by the PL7 viewer:



Description of display fields

The following table describes the different display fields:

Field	Description
Release	The system error messages are without acknowledgement. This zone always shows "Without acknowledgement".
Error	Shows the language object which detected the error
Zone	ALWAYS shows at 0 (system zone)
Appearance	Date of the appearance of the error
Disappearance	Date of the disappearance of the error
Message	Displays The Error Message
Status	The errors generated by the diagnostics system have no status word.

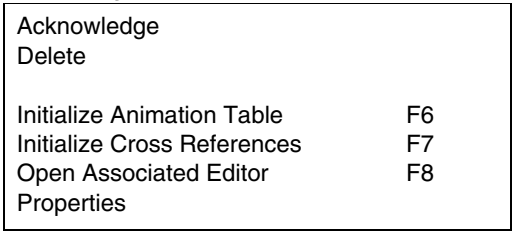
The messages appear at the end of the list.

Additional error message display functions by the PL7 viewer.

Introduction	As well as the message display function, the viewer in PL7 provides access via a contextual menu to additional functions which can be used to identify faults in more detail.
How to access the contextual menu	The contextual menu may be accessed by right clicking on the error message.

Illustration

The following illustration shows the contextual menu:



Commands description

The following table describes the different commands:

Command	Description
Acknowledge	Nonactive (the system error messages are without acknowledgement)
Delete	Deletes the error message from the viewer. **Note:** It is not possible to delete the message concerning an error which has not disappeared
How to initialize an animation table	Display an animation table which contains the values of bits and system words that detected the error.
How to initialize cross references	Display an animation table of cross references relating to the bits and system words that detected the error.
Opening the associated editor	Views the program element where the error has been detected, or opens the configuration editor on: • bacs configuration screen for a local input/output fault, • FIPIO configuration screen for a FIPIO input/output fault.
Properties	Displays a dialog box which contains detailed information on the detected fault.

Introduction to the advanced Properties dialogue box associated with the diagnostics system

General

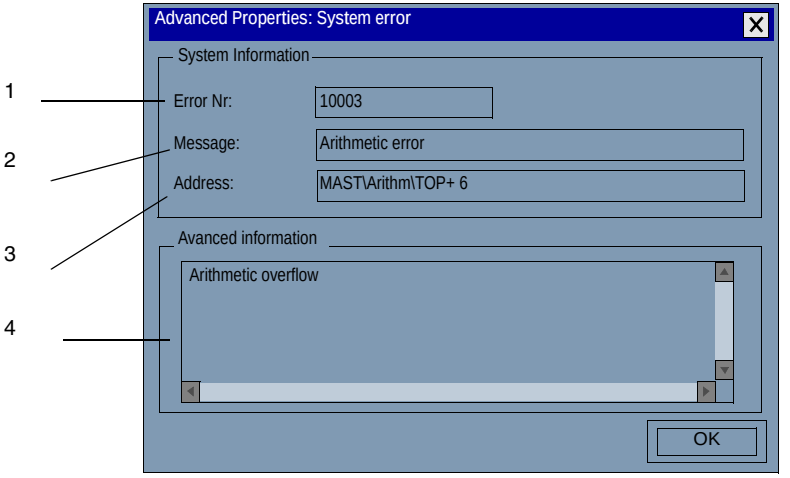
This dialogue box provides more detailed information on the fault detected.

To access the dialogue box:

- double click on the message
- press the **Enter** key
- or activate the **Properties** command from the contextual menu.

Operations screen

The screen below shows the advanced properties dialogue box:

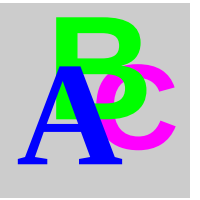


Description

The following table describes the different dialogue box areas.

Address	Description
1	Error number (see list in *Introduction to the diagnostics system, p. 132*)
2	Error message
3	Program address where the error is located
4	Detailed description of the error

Index



N

P

R

S

T

V