SIEMENS

SIMATIC

S7-PDIAG for S7-300 and S7-400 Configuring Process Diagnostics

Manual

Edition 07/2005
A5E00473485-01

Safety Guidelines

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring to property damage only have no safety alert symbol. The notices shown below are graded according to the degree of danger.

Danger
indicates that death or severe personal injury **will** result if proper precautions are not taken.

Warning
indicates that death or severe personal injury **may** result if proper precautions are not taken.

Caution
with a safety alert symbol indicates that minor personal injury can result if proper precautions are not taken.

Caution
without a safety alert symbol indicates that property damage can result if proper precautions are not taken.

Attention
indicates that an unintended result or situation can occur if the corresponding notice is not taken into account.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

Qualified Personnel

The device/system may only be set up and used in conjunction with this documentation. Commissioning and operation of a device/system may only be performed by **qualified personnel.** Within the context of the safety notices in this documentation qualified persons are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

Prescribed Usage

Note the following:

Warning
This device and its components may only be used for the applications described in the catalog or the technical description, and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens.
Correct, reliable operation of the product requires proper transport, storage, positioning and assembly as well as careful operation and maintenance.

Trademarks

All names identified by ® are registered trademarks of the Siemens AG.
The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Siemens AG
Automation and Drives
Postfach 4848
90437 NÜRNBERG
GERMANY

A5E00473485-01

Preface

Purpose of the Manual

This manual gives you a complete overview of working with S7 PDIAG. It helps you during installation and commissioning. The procedures involved in creating programs, the structure of user programs and the individual language elements are described.

This manual is intended for the programmers of STEP 7 programs and for those responsible for configuring, commissioning, and servicing automation systems.

We recommend you get acquainted with the example in the chapter "User Guide for S7 PDIAG". It offers you an easy start in the configuration of a process diagnostics.

Required Basic Knowledge

You require a general knowledge in the field of automation engineering to be able to understand this manual.

In addition, you should know how to use computers or devices with similar functions (e.g. programming devices) under MS Windows 2000, MS Windows XP and MS Windows Server 2003 operating systems. Since S7-PDIAG is based on the STEP 7 software, you should also know how to operate it. This is provided in the manual "Programming with STEP 7 V5.3".

Where is this Manual valid?

This manual is valid for the software package S7 PDIAG V5.3.

You will find information on service packs which may be developed after the book has gone to print:

- in the "Readme.wri" file
- in the updated S7-PDIAG Online Help

The topic "What's new?" in the Online Help offers you a good starting point and an initial overview of the innovations in S7-PDIAG version V5.3.

Place of this Documentation in the Information Environment

This manual forms part of the S7 PDIAG V5.3 documentation package.

The documentation package "STEP 7 Basic Information“ offers you, as the title says, basic knowledge on STEP 7.

The following table displays an overview of the STEP 7 documentation:

Documentation	Purpose	Order Number
STEP 7 Basic Information with • Working with STEP 7 V5.3, Getting Started Manual • Programming with STEP 7 V5.3 • Configuring Hardware and Communication Connections, STEP 7 V5.3 • From S5 to S7, Converter Manual	Basic information for technical personnel describing the methods of implementing control tasks with STEP 7 and the S7-300/400 programmable controllers.	6ES7810-4CA07-8BW0
STEP 7 Reference with • Ladder Logic (LAD)/Function Block Diagram (FBD)/Statement List (STL) for S7-300/400 manuals • Standard and System Functions for S7-300/400	Provides reference information and describes the programming languages LAD, FBD, and STL, and standard and system functions extending the scope of the STEP 7 basic information.	6ES7810-4CA07-8BW1

Online Helps	Purpose	Order Number
Help on STEP 7	Basic information on programming and configuring hardware with STEP 7 in the form of an online help.	Part of the STEP 7 Standard software.
Reference helps on STL/LAD/FBD Reference help on SFBs/SFCs Reference help on Organization Blocks	Context-sensitive reference information.	Part of the STEP 7 Standard software.

Further Support

If you have any technical questions, please get in touch with your Siemens representative or agent responsible.

You will find your contact person at:

http://www.siemens.com/automation/partner

You will find a guide to the technical documentation offered for the individual SIMATIC Products and Systems here at:

http://www.siemens.com/simatic-tech-doku-portal

The online catalog and order system is found under:

http://mall.automation.siemens.com/

Training Centers

Siemens offers a number of training courses to familiarize you with the SIMATIC S7 automation system. Please contact your regional training center or our central training center in D 90327 Nuremberg, Germany for details:

Telephone: +49 (911) 895-3200.

Internet: http://www.sitrain.com

Technical Support

You can reach the Technical Support for all A&D products

- Via the Web formula for the Support Request
  http://www.siemens.com/automation/support-request
- Phone: + 49 180 5050 222
- Fax: + 49 180 5050 223

Additional information about our Technical Support can be found on the Internet pages http://www.siemens.com/automation/service

Service & Support on the Internet

In addition to our documentation, we offer our Know-how online on the internet at:

http://www.siemens.com/automation/service&support

where you will find the following:

- The newsletter, which constantly provides you with up-to-date information on your products.
- The right documents via our Search function in Service & Support.
- A forum, where users and experts from all over the world exchange their experiences.
- Your local representative for Automation & Drives.
- Information on field service, repairs, spare parts and more under "Services".

Contents

1 Product Overview

1.1 Process Diagnostics with S7-PDIAG

S7-PDIAG enhances the function scope of the STEP 7 standard software with process diagnostics functions for the programming languages LAD/STL/FBD. The Process Diagnostics function detects errors in the user process (manufacturing, distribution, processing, etc.) and acquires information about:

- The type of error
- The location of the error
- The cause of the error in your process.

S7-PDIAG also provides troubleshooting information on the display unit (HMI).

Process Diagnostics Tasks

The process diagnostic function monitors production systems. The basic operating principle of this monitoring system is quite simple. It compares the actual and setpoint status of process signals. In this context, the task could use a watchdog function, for example (e.g. the permitted adjustment time for a valve) or perform a plausibility check (e.g. the status of both high/low limit signals of a valve must not be "1" at the same time). Such control mechanisms allow you to check the entire signal sequence, ranging from the instruction set to the output channel, terminals, cables, actuators, limit switches and back to the input channel.

Typical errors are, for example:

- Runtime errors at actuating elements
- Motor protection tripped
- Limit switch fault
- Interlock condition not met...

If the user decides to program monitoring functions without system support, the volume of this user program may well reach the scope of the control program. Keep in mind that you would also have to configure corresponding error messages for the display devices (HMI). When you modify the control program you usually need to reconfigure the monitoring functions also, which by itself is a time-consuming task.

In those situations our optional software packages such as S7-PDIAG and ProAgent offer support for the configuration of process diagnostics functions and their visualization. The functions of this software substantially reduce engineering time required for programming monitoring and message functions. You merely need to perform a further generating run, since this system automatically corrects the monitoring functions after control data has been modified.

Competitive strength

System and process operators continually need to minimize production costs in order to remain internationally competitive in industry. Down times in production systems can lead to loss of production and therefore represent an important cost factor. Process diagnostics with S7-PDIAG is aimed at the reduction of this cost factor.

Customer Benefits

You can, of course, recognize process errors without using S7-PDIAG. S7-PDIAG, however offers more efficient ways and means of designing plant-specific processes and of configuring special functions for monitoring "neuralgic points" in your process. The advantages are:

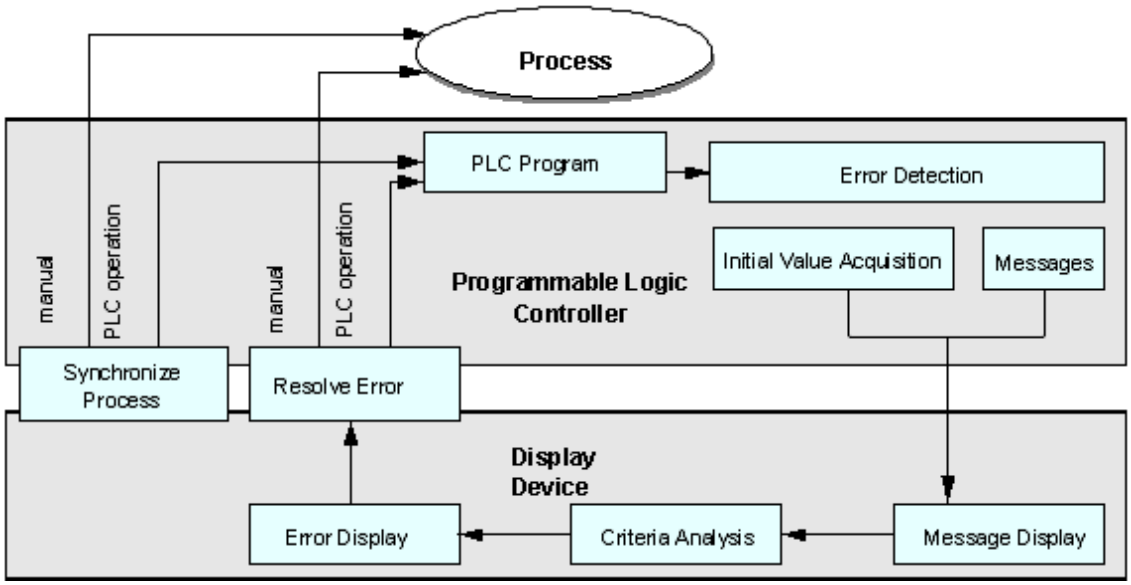
- S7-PDIAG detects process runtime errors at an early stage, and thus significantly reduces down times and production losses.
- Efficient troubleshooting routines due to detailed error information which is also available to the machine operator. After an error has occurred in your control hardware or production system, the SIMATIC system and process diagnostics functions immediately provide you with corresponding plain text error messages, including the date and time.
- Several errors occurring in short succession are displayed in the correct time order. You can also view further details, if available.
- Simple and fast configuration of objects and methods, as well as minimum programming effort for motion monitoring.
- The system provides detailed information (e.g. b means of a criteria analysis), and thus supports troubleshooting routines via the operator station (HMI), without any call for additional programming.
- Consistency between the diagnostics and the user program

1.2 Overview of Functions

Sequence of Process Diagnostics Functions

Process diagnostics is split into three topics: error recognition, error display and troubleshooting.

When an error occurs in a user process, the following process diagnostics functions will be triggered:



1. **Error recognition:**
   The error recognition function is performed on an S7 CPU via the blocks generated by S7-PDIAG. When an error has occurred, the system reads and saves the error signal status of the corresponding addresses (initial value acquisition), and provides this information for subsequent display and analysis. It also reports the error ID and associated values (e.g. temperature reached) to connected HMI operator panels as required.

2. **Message display:**
   S7-PDIAG recognizes incoming and outgoing errors and displays them at the HMI. The two criteria for displaying this information on the HMI are:

   - A message window displays the error message in plain text, including the date and time.
   - A detail window provides additional information to the user, i.e. the relevant signals, including the logic code in LAD or STL. A criteria analysis determines and indicates the triggering signals.

3. **Criteria Analysis:**
   By means of a criteria analysis based on the initial values, S7-PDIAG in conjunction with the operator panels can be used to locate the error triggering address and thus the cause of the process error (with Boolean program logic).

4. **Troubleshooting:**
   Troubleshooting routines can be performed either by user intervention in the process and/or by manual operator control via the control system (at the operator panel). Since troubleshooting usually requires operator control of aggregate motions, a standardized motion screen is implemented to support such routines. The user can also toggle auto and manual mode at the HMI.
5. **Restarting the Process:**
   Depending on the type of error eliminated, the user can decide to continue operation directly or initialize the system before he resumes operation.

1.3 Process Monitoring

S7-PDIAG allows you to monitor specific process error events. You can configure relevant error definitions while you create the user program, or you can do so at a later stage.

Available monitoring modes:

- **Address Monitoring**
  lets you direct the monitoring specifically to level changes or edge transitions at certain addresses. This function can be combined with a delay time function. You can monitor addresses without having to modify your user program.
- **Motion Monitoring**
  lets you monitor the correct execution and speed of physical motions in your process. The motion monitoring concept requires of you to follow the programming conventions and thus to adapt your user program.
- **Global Monitoring**
  lets you monitor process errors which are the result of logic operations with multiple addresses, without the need to modify your user program. S7-PDIAG will only generate an error message if this logic condition has been met.

Error Definition

You specify exactly which error event is to be monitored by means of an error definition in an address, motion or general monitoring definition. S7-PDIAG also allows you to monitor all addresses of the type BOOL.

Monitoring Blocks

Based on the error definitions configured, S7-PDIAG generates monitoring function blocks you need to download to the CPU in order to be able to monitor your process.

1.4 When an Error Occurs...

When S7-PDIAG detects an error, it transfers the user-specific error message text to all connected HMIs during runtime of the user program. You can also input message texts and associated values when you configure the error definitions.

You can assign each message a priority class (1 to 16). In this way, you can react specifically to errors of different priority classes in your user program.

Initial Diagnostic Address (IDA)

The initial diagnostic address (IDA) represents a tag at which error tracking starts, provided a criteria analysis is to be performed. This address location must contain either an assignment or a "Set" or "Reset" operation.

Initial Value Acquisition

If you have set initial value acquisition, all initial values of the monitored address are saved in the PLC within the same cycle in which the error was detected. Initial values are binary states that form the result of a logic operation at the monitored address.

The information on which addresses are involved and how these are interconnected is fetched directly from your user program.

Criteria Analysis

A criteria analysis is performed on the operator panel in order to analyze the error conditions. A criteria analysis can be performed only for boolean addresses (see "Readme.wri") and starts at the initial diagnostic address (IDA). Criterial analysis evaluates the initial values of all networks which determined the value of the initial diagnostic address. You can then display the states of the addresses (initial values) which caused the error (for example, limit switch at input I1.1) up to the result of logic operation in STL, LAD, and FBD directly on the operator panel.

Positive criteria analysis assumes the signal state "1" of the initial diagnostic address to be correct, while negative criteria analysis assumes that the signal state "0" is correct.

The criteria analysis is available for all boolean input parameters of a function block and can thus be performed across the block limits.

Exclusion Addresses

You can create a list of so-called "exclusion addresses" for criteria analysis, which are defined as "never causing an error". Criteria analysis then hides these addresses and the subnets in which they are found, if they have been registered as having the value "0" (this is only possible in conjunction with ProAgent, version 5.0 or higher). This allows a differentiation between auto and manual mode, for example.

Termination Addresses

You can also define addressees referred to as "termination addresses" in a list. The system terminates the use of auxiliary networks for criteria analysis when this termination address is reached.

Associated Values in Message Texts

An associated value represents a value (or address) which can be "added" to a message text. This value is recorded by S7-PDIAG at the time the error is detected. The associated value is displayed by the display system at the position in the message text which you have specified. Enter the corresponding formal address in the message text.

This associated value can be a parameter of the type BOOL, BYTE, CHAR, WORD, INT, DWORD, DINT and REAL of the I, Q, M or DB area. You can specify the position and the format of the associated value in your message text.

Status/Actual Value Acquisition

Using the status/actual value acquisition of the initial value addresses, you can check on the display device whether an error status has actually been resolved (this is only possible in conjunction with ProAgent, version 5.0 or higher).

Group Error Bit ID

There is a group error bit ID in each of the user data types Unit," "S_UNIT," and "Motion." After an error has occurred, S7-PDIAG sets this bit in all primary units and motions. An error in the subordinate unit "Chuck" will thus also appear in the primary unit "Drill".

Mandatory Message Acknowledgement

Mandatory message acknowledgement can be configured separately for specific messages. You thus determine whether the operator is obliged to read and manually acknowledge this message at the operator panel or whether the message is allowed to be cleared "without having been read" after the error has been resolved.

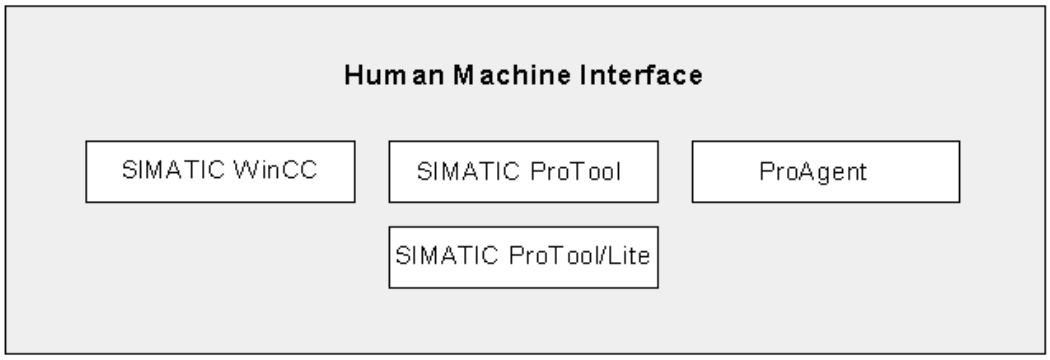
Modifying Times in Monitoring Definitions

You can "modify the monitoring time" (not equal to 0) in existing monitoring definitions without having to generate the monitoring blocks again. Since you can do so both online and offline, you have the advantage that you can determine the monitoring time step- by-step.

1.5 Human Machine Interface (HMI)

Human Machine Interface is a software designed especially for SIMATIC operator control and monitoring functions.

- The open process visualization system SIMATIC WinCC is an independent standard system that provides all essential operator control/monitoring functions for engineering systems and applications.
- SIMATIC ProTool and SIMATIC ProTool/Lite are modern tools for the configuration of SIMATIC Operator Panels and SIMATIC C7 compact units.
- ProAgent provides functions for directed and fast process diagnostics in plants and machinery by means of acquisition of data relevant to the location and cause of errors.



1.6 Configuration of Process Diagnostics

In S7-PDIAG, process diagnostics can only be performed in interaction with an operator panel (HMI) and a corresponding software such as ProAgent for ProTool/Pro or WinCC.

Configuration

In the first step, you program the PLC functions.

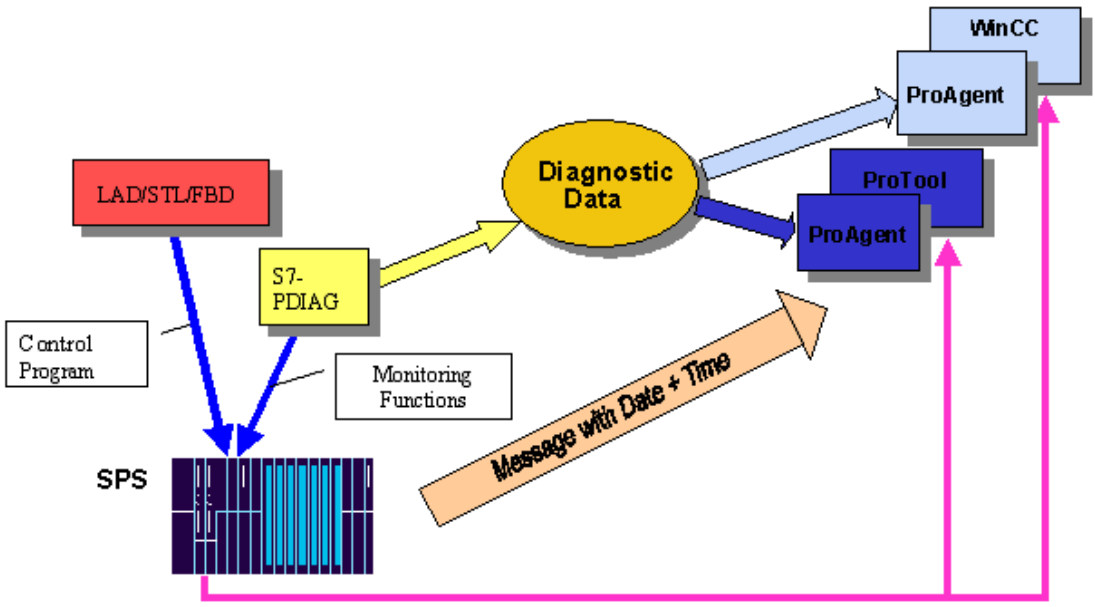
These include:

- Programming control functions under LAD/FBD/STL,
- Defining/programming monitoring function under S7-PDIAG,
- Compilation of the control program and generating diagnostics functions, and
- Download of the blocks to the AS.

In the second step, you configure the HMI function parameters.

These include:

- Selecting the default images provided by ProAgent,
- Selecting the controls whose signals are to be displayed by the HMI system,
- The number of plant units displayed,
- Generating the HMI project and
- Download of the compiled data to the display unit (HMI).



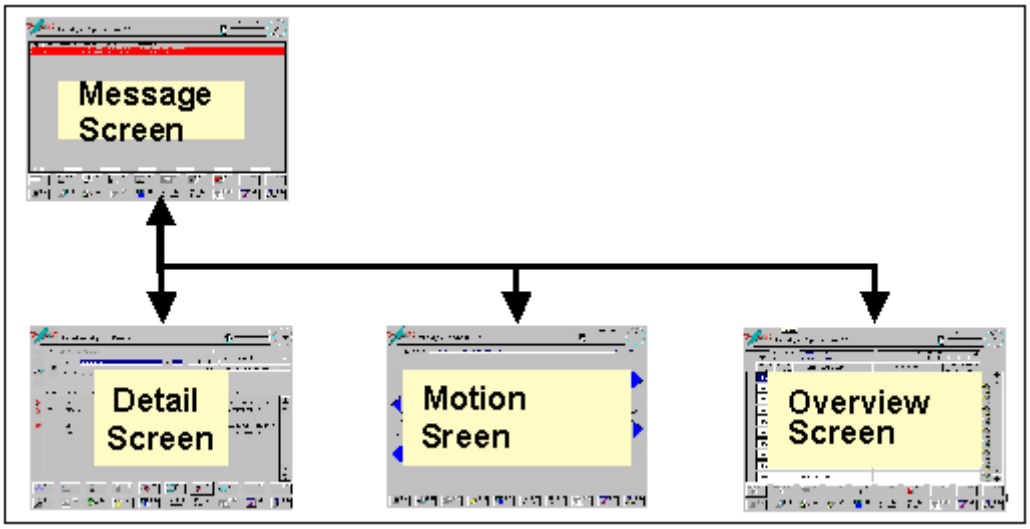
1.7 Troubleshooting at the Display Device (HMI)

S7-PDIAG returns information for supporting troubleshooting to the operator panels (HMI). On the operator panel you can use the criteria analysis to analyze the conditions which led to a process error. In this case you use the initial values stored in the PLC.

Corresponding configuration data are provided to the display units by means of the configuration software **SIMATIC WinCC, SIMATIC ProTool, SIMATIC ProTool/Lite** and **ProAgent**.

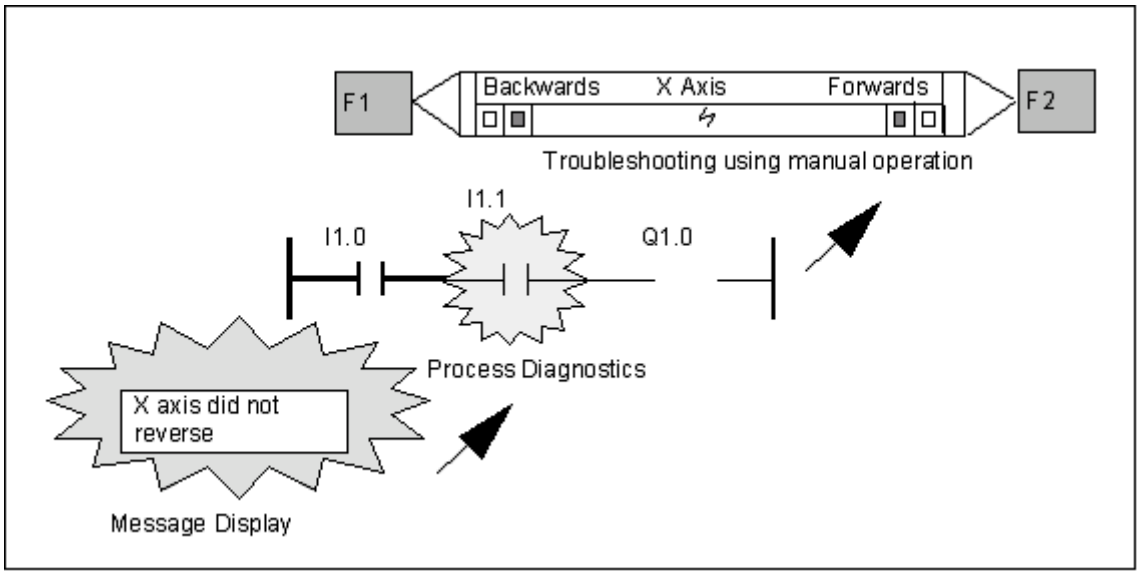
Diagnostic Screens on the Display Device

The display units (HMI) show diagnostic information in four different images which ProAgent provides by default and which you can select as required:



1. The Message Screen displays all queued error messages, including the time and date,
2. The Diagnostics Detail Screen displays the result of a criteria analysis relevant to an error message. This analysis determines the error message triggering signals in the user program and marks the signals that cause the error (criteria analysis).
3. The Motion Screen displays all executable motions of a unit and options for operator control of specific aggregates via the arrow keys.

4. The Diagnostics Screen displays the current status of all system units.

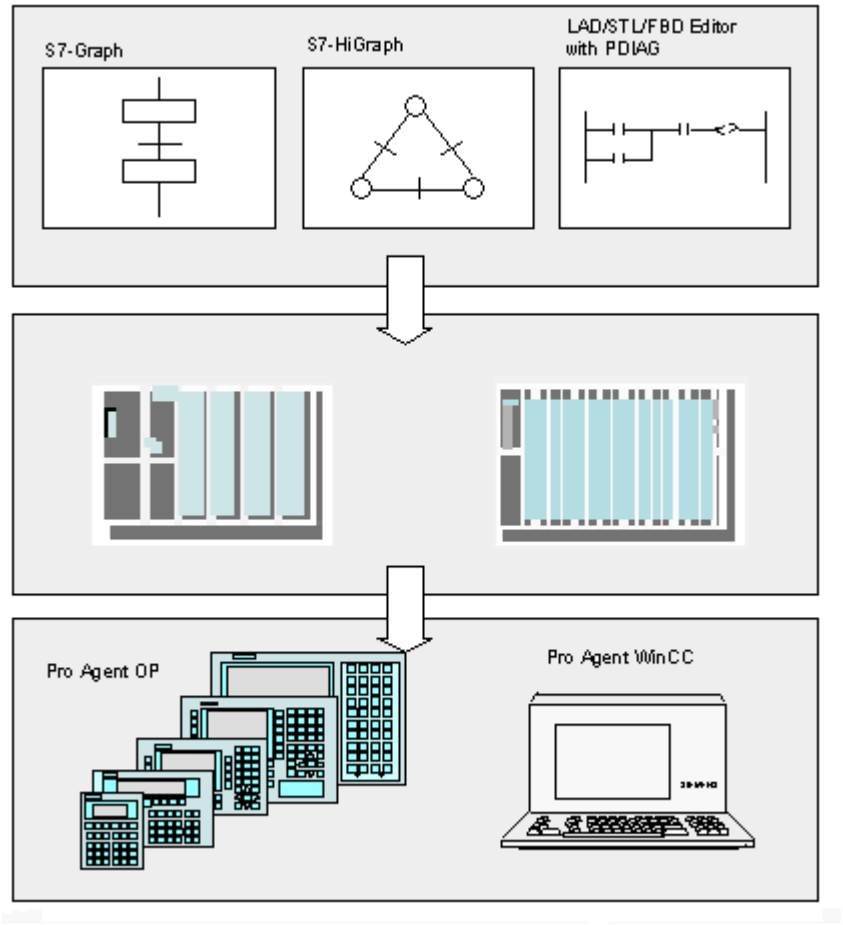


For further information refer to the documentation of your configuration software (e.g. SIMATIC WinCC, SIMATIC ProTool and SIMATIC ProTool Lite or ProAGENT).

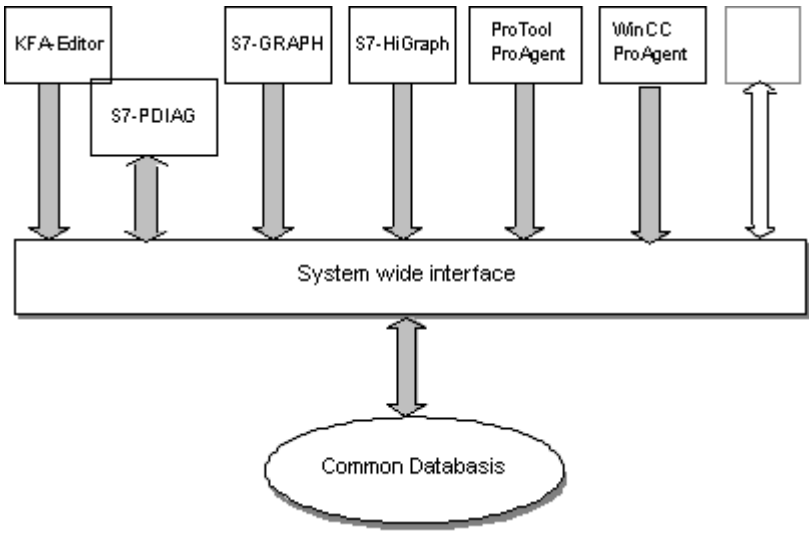
1.8 System-wide Interaction

S7-PDIAG Process Diagnostics is fully incorporated into the SIMATIC S7 configuration software. You can configure diagnostic data while you are implementing your user program.

The figure below shows how all the components of process diagnostics interact throughout the system.



These, as well as all further data relevant for process diagnostics are stored in a common database as shown in the figure below.



This incorporation into the global system significantly reduces expenditure for programming process diagnostics functions.

1.9 S7-PDIAG Documentation

The full S7-PDIAG documentation is found in the Online Help. The Online Help is split into two section:

- Immediate, context-sensitive help is available on the basis of Windows functions.
- The Online Help provides global information in HTML format. You can access this information by calling **Help > About**, regardless of the currently opened application.

This Online Help is supplemented with an electronic manual in PDF format. You can open this documentation via the task bar under **Start > Simatic > Documentation**.

2 Installation and License Management

2.1 Automation License Manager

2.1.1 License management with Automation License Manager

The use of your the programming software requires a product-specific license key (authorization). Starting with S7-PDIAG V5.3, this key is installed using Automation License Manager (ALM.)

Automation License Manager is a software product of Siemens AG, and is used to manage the License Keys (technical representative of licenses) of all systems.

Automation License Manager is available on:

- the S7-PDIAG product CD
- the Internet pages of A&D Customer Support of Siemens AG as Web Download.

Automation License Manager features an integrated Online Help. You can open this help after you installed ALM by pressing F1, or selecting **Help > Help on License Manager**. This help system provides detailed information about the functionality and handling of Automation License Manager.

Licenses

STEP 7 program packages are legally protected by license agreements. You purchase a license that entitles you to legally use a product. Representatives of those legal rights are:

- the CoL (**C**ertificate **o**f **L**icense), and
- the License Key.

Certificate of License (CoL)

The "Certificate of License" provides evidence of the legal right of use, and is issued with the product. This product may only be used by the licensee or his representatives.

License Keys

The License Key is the technical representative of a license (electronic license stamp.)

SIEMENS AG issues a License Key for its software products which are legally protected by license. The software product can only be used if a valid License Key is found on the computer, and in compliance with the license agreement an terms of use associated with this License Key.

Notes

- To get you started with the user interface and functionality of your S7-PDIAG software, you may be run in demo mode without License Key over a short period.
- However, the unrestricted use is only permitted and possible with License Key, in compliance with license agreements.
- If you have not installed the license key, you will be requested to do so at regular intervals.

License Keys may be stored on the media outlined below, and may be transferred between those:

- License Key disks
- Local hard disk drives
- Network drives

For further information about the handling of License Keys, refer to the ALM Online Help.

License categories

The licenses for software products of Siemens AG are distinguished based on the application-oriented categories outlined below. The behavior of a software is determined by its License Key category. The category of use is derived from the relevant CoL.

License category	Description
Single License	Allows legal and unrestricted use on any computer.
Floating License	No time limit; allows legal use of a software product based on a network license ("remote" use.)
Trial License	Limits use of the software to:· • a period of max. 14 days, • a certain number of days, beginning with its initial use, • test and validation purposes (exclusion of liability.)
Upgrade License	An upgrade may require a specific system state: An Upgrade License can be used to migrate from an "old" version x to a new version >x+... May also require expansion of the data volume capacity.

2.1.2 Installing Automation License Manager

Automation License Manager is installed by means of a Setup routine included on the product CD of S7 - PDIAG.

You can install the Automation License Manager alongside with S7-PDIAG or at a later time.

Notes

- For detailed information about the installation of Automation License Manager, refer to its current "Readme.wri" file.
- The ALM Online Help contains all necessary information about the functionality and handling of License Keys.

Subsequent installation of License Keys

When you run the software and a License Key is not found, the system outputs a message warning you of this state.

Notes

- You may use the standard software without a license key over a short period, in order to obtain an initial overview of its user interface and functionality.
- However, unrestricted use of the software in accordance with the license agreement license requires installation of the License Key.
- If you have **not** installed the License Key, you will be asked to do so at regular intervals.

License Key installation options:

- Installation from License Key disks
- Installation of License Keys via Web Download (please order)
- Use of Floating License Keys on the network.

For detailed information on installing License Keys, refer to the ALM Online Help. To open this help, press F1 or select the **Help > Help on License Manager** command.

Notes

- In Windows 2000/XP/Server 2003, License Keys can only be used on a hard disk drive that is not write protected.
- Floating Licenses may also be used on a network, i.e. "remotely."

2.1.3 License Key guidelines

Caution

Always follow the License Key guidelines in the Online Help and readme.wri of ALM. If ignored, you risk irrevocable loss of your License Keys.

You can open the ALM Online Help either context-sensitive by pressing F1, or by selecting **Help > Help** on Automation License Manager.

This help system provides all relevant information about the functionally and handling of License Keys.

2.2 Installing S7-PDIAG

2.2.1 Requirements of S7-PDIAG

Hardware requirements

- S7-PDIAG requirements in analog to the STEP 7 standard package. Additional requirements:
- Your S7 CPU must contain SFC17 and SFC18, otherwise S7-PDIAG will be unable to process Alarm_S, and an error message will appear while downloading the blocks.
- S7-PDIAG requires additional memory space. Refer to the current readme file.

Software requirements

S7-PDIAG V5.3 runs on a PC/PG under:

- the operating systems Microsoft Windows, Windows 2000, Windows XP, Windows Server 2003.
- STEP 7 standard software, version 5.3.

Memory requirements of the S7 CPU

Memory requirements of your S7 CPU are determined by the

- error detection and
- initial value acquisition

components.

Memory requirements of error detection

Basic requirement	approx. 1440 bytes
Per monitoring function, without timer	approx. 8 to 12 bytes
Per monitoring function, with timer	approx. 104 to 110 bytes
Per monitoring function, with associated value	approx. 26 to 36 bytes

The above standard values are determined by the complexity of the monitored networks and the corresponding monitoring logic.

Memory requirements of initial value acquisition

Basic requirements	approximately 1470 bytes
additional for configured memory word	220 bytes
per monitoring function	approximately 22 bytes
per address	approximately 4 bytes

Additional AS memory for network data, relevant for diagnostics:

Basic requirements	168 bytes
per monitoring function	8 bytes
per operator	1 byte

The above standard values are determined by the complexity of the monitored networks and the corresponding monitoring logic.

2.2.2 Installing S7-PDIAG

Setup installs S7-PDIAG automatically. The installation routine is menu controlled. Setup is called via the standard Windows 2000/XP/Server 2003 software installation dialogs.

The essential phases of installation are:

- Copying the data to your engineering system
- Authorization (if required).

Preparations

Basic requirements is the installation of Windows 2000/XP/Server 2003 and of the STEP 7 standard software.

Running Setup

Procedure:

1. Open the Windows 2000/XP/Server 2003 "Control Panel". Click "Software" to open the installation dialog
2. Click "Install"
3. Insert the data medium, then click "Continue." Windows 2000/XP will automatically locate SETUP.EXE.
4. Follow the instructions of Setup

A message will be displayed on the screen to inform you that Setup has been completed successfully.

Installing License Keys

Setup checks the existence of a corresponding License Key on the hard disk drive. If a valid License Key is not found, the system outputs a message indicating that the software will only run in demo mode if the License Key is missing. Continue Setup by installing the License Key now, or install it at a later time. In the first case, insert the included License Key disk in your drive when asked to do so.

2.3 Uninstalling S7-PDIAG

Use the general uninstall routine of Windows:

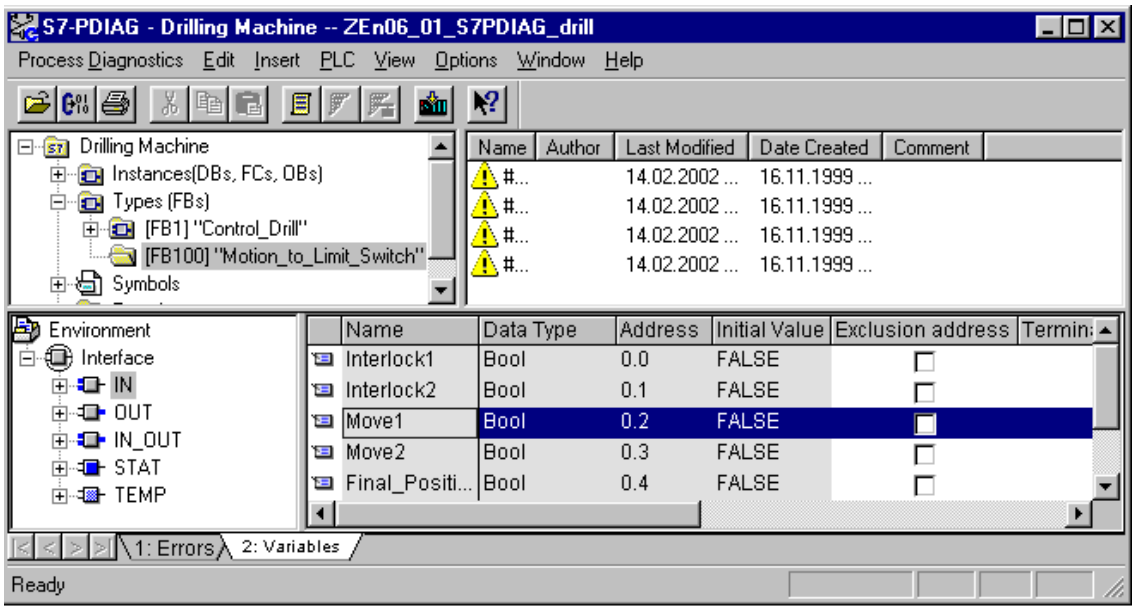
1. Run the software installation dialog by double-clicking the "Software" icon on the "Control Panel".
2. Select S7-PDIAG from the list of installed software, then click "Remove" to uninstall it.

3 Introduction to S7-PDIAG

3.1 User Interface

3.1.1 Unit Overview

The unit overview represents the user interface of S7-PDIAG. S7-PDIAG automatically opens the unit overview on startup.



Left Window Partition

The tree structure displayed in the left-hand partition of the unit overview represents the hierarchy of the individual unit and motion objects and shows the directories created for:

- Instances (with group folders),
- Types,
- Symbols and
- Templates.

By double-clicking an object, you can display further levels in the structure. You can also display the properties of an object using the right mouse button.

Right-hand Window Partition

On the right-hand side of the unit overview, a list is displayed containing the existing error definitions for the object which has been selected on the left. This section of the unit overview also shows additional information on each error definition (author, last modification, date created, comment). If you select an error definition and then click the right mouse button, a list of different editing functions appears for you to choose from.

The unit overview is automatically updated when you compile your error definitions into monitoring blocks.

'Details' Window

The 'Details' window of S7-PDIAG is positioned at the bottom edge of the unit overview.

- The 1:Error tab displays information on errors occurred, e.g. when generating data,
- The 2:Variables tab is used for editing variables and block parameters.

You can here directly edit the interface of the block selected and define break points and exclude addresses. Further information can be called by pressing the F1 key.

Display of Units Moved

You can move units in order to create a structure for objects in the left partition of the unit overview according to your system requirements. You can thus, for example group all objects belonging to the same machine, irrespective of the block in which they actually exist.

Moved units are identified by a special symbol. You can always undo the move operation via the menu command **Edit > Undo Move** or the corresponding context-sensitive menu.

Display of the Virtual Unit Overview

After having moved units, you can click on menu command **View > Virtual Unit Overview** to open a virtual overview of these units. Click on this men command again to exit this view.

3.1.2 Units

Units structure the process view according to components which are related to one another by their technical function. If you have set up your project so that each block relates to a physical object in the process (for example, a press, a stamp, or a cover), the units represent your process image. Your program will thus contain a unit for each of its blocks assigned the diagnostics system attribute.

Units can be monitored by means of error definitions. Blocks assigned the "S7_pdiag = true" system attribute can always form a unit.

System Structure

Units can also store global data for all nested objects of the hierarchy (e.g. operating modes and group error bits). A unit may contain error definitions, motions, and other subunits. Using units, you can combine both individual errors and motions into a system unit.

Advantages of the Unit Structure of Processes

A unit structure of system processes enables you to localize process errors precisely, since the ProAgent display units show each unit assigned the diagnostics attribute on a separate overview screen. The diagnostic function can thus quickly localize the unit, i.e. the plant or machine section that has caused the error.

Real and Moved Units

Units are represented with other objects in a tree structure in the unit overview. You can move units in order to display them in a virtual unit overview. This function is useful for improving the process structure. Units moved are identified by a special symbol.

Units for a data block, function, or organization block are also visible in the unit overview screen on the display devices.

Grouping Units

In addition to the default standard group you can group any number of units in up to 15 different groups. You should, however group the units after you have created the structure, i.e. after you have completed your program hierarchy.

3.1.3 Motions

A motion is a sequence in the process; for example, a punch moving up and down. Such motions can be monitored by means of error definitions.

Motions in a process are often defined as follows:

- They have two directions with two or more stable final positions
- They can be moved in the corresponding direction when triggered

For example, a cylinder moves from the current final position to the target final position when the hydraulic pressure is switched on.

By using UDT_Motion and the LAD networks for motion programming included with S7-PDIAG, you can display and control motions without the need of further programming using the motion screens of the display unit.

System Structuring

A plant or machine may contain a large number of motions. It is therefore a good idea to combine motions with similar functions together in a subsystem, which is referred to here as a unit.

You can create several error definitions for each motion. A motion can only exist in units, and represents an actual motion of a physical object in the process (for example, a punch die moving up and down).

Layout

Motions are represented with other objects in a tree structure in the unit overview. When you create your instances, the new motions are depicted in the unit overview.

Motions displayed in the unit overview are output and can be controlled by the user on the motion screens of the display devices (using ProAgent). This information may include actual limit position values, for example.

3.1.4 'Details' Window

The 'Details' window of S7-PDIAG is positioned at the bottom edge of the Unit Overview and displays error messages and variables.

Here you can define break points and exclude addresses for your FBs.

1:Error tab

Displays error messages relevant to S7-PDIAG.

2:Variables tab

Displays the interface of the block selected from the tree structure of the unit overview as well as the corresponding variables and block parameters.

The variables tab is split into two partitions: the left shows the variable overview, the right shows the variable detail view.

- In the variable overview, you can edit variables and block parameters.
- The variable detail view provides detailed information on declared block parameters and variables. The display in some columns depends on the object type to be edited, the selected declaration segment or on the selected variables. You can here declare break points and exclude addresses for process diagnostics. You can call further context-sensitive information by pressing the F1 key.

Displaying the 'Details' Window

You can open and close the 'Details' window of S7-PDIAG via the **View > Details** menu command and change its size as required.

3.2 Basics of Process Diagnostics

3.2.1 Initial Diagnostic Address (IDA)

The initial diagnostic address is the address to which the error definition in the block is linked. Error tracking starts at this address when a criteria analysis is performed. The address must either contain an assignment or a "Set" or "Reset" operation

Requirements for Creating Error Definitions

Before you create an error definition, you must first select the initial diagnostics address.

3.2.2 Error Monitoring

S7-PDIAG allows you to monitor specific process error events. You may configure these errors either while you are creating your user program or you do so at a later stage. Monitoring modes available are:

- **Address Monitoring**
  lets you direct monitoring specifically towards level changes or edge transitions at certain addresses. The function can be combined with a delay time function. You can monitor addresses without having to modify your user program.
- **Motion Monitoring**
  lets you monitor the correct execution and speed of physical motions in your process. The requirement for motion monitoring is that you follow the programming conventions and adapt your user program.
- **Global Monitoring**
  lets you monitor process errors which are the result of logic operations with multiple addresses, without the need to modify your user program. S7-PDIAG will only generate an error message if this logic condition has been met.

Enabling Error Monitoring

You first need to enable the error monitoring function in order to use its definition by clicking on the "Enable Monitoring" check box after you have created the error definition.

If you have activated error monitoring, the error definition is compiled to form a monitoring definition in the error recognition block. This enables the error to be detected in the active process and results in a corresponding message on the display device.

Note

If you do not enable error monitoring, the error definition still remains stored in the database of the project, but no monitoring definition is generated in the error-detection block on compilation. This means that no errors can be detected or recorded during the process. In this way, you can temporarily disable error monitoring.

3.2.3 Initial Value Acquisition

When initial value acquisition is enabled for an error definition, the function records all binary states of addresses used to form the monitored address (IDE) during the cycle in which an error was detected. You can use these to perform a criteria analysis and thus simplify troubleshooting routines.

Click on the "Initial Value Acquisition" check box to enable this function for an error definition at the time you declare or modify the definition.

Prerequisite for Criteria Analysis

Initial value acquisition must be enabled for criteria analysis at the display unit after an error has occurred. This makes it easier to evaluate and find the location of the error.

Notes

- You can only enable initial value acquisition if the "Monitoring Active" check box has also been set for the corresponding error definition.
- On compilation, you can disable the option of generating data for initial value acquisition for the project via the menu command **Options > Customize**.

3.2.4 Criteria Analysis

Criteria analysis uses the user program logic to determine the cause of an errors, is performed on the operator panel and used to track the cause of errors. The criteria analysis starts at the initial diagnostic address and evaluates the initial values of all networks which determine the value of the initial diagnostic address.

Positive criteria analysis assumes the signal state "1" of the initial diagnostic address to be correct, while negative criteria analysis assumes that the signal state "0" is correct.

Requirement

Errors detected can only be evaluated by means of a criteria analysis if initial value acquisition is enabled.

A criteria analysis is performed in two steps:

- First of all, it determines the RLO values based on the initial values for all addresses in the network in which the error occurred.
- Secondly, it checks each row is and marks it as correct or incorrect, starting at the end of the network.

Result of a Criteria Analysis

Identification of all rows of the examined network which contribute to the error.

Notes

- If you have defined exclusion addresses, the subnets will be ignored ion this case.
- At the FCs, it is not possible to track errors beyond block boundaries by means of the criteria analysis. In this case, the call parameters will not be transferred to these blocks via static data as would be done in the case of FBs and the corresponding instances. It is therefore not possible to access these data later and **not** possible to track these parameters beyond block boundaries during criteria analysis.

3.3 Assigning Diagnostic Functions to Blocks

You need to create the data S7-PDIAG requires for the selected block in order to perform process diagnostic operations. This is done automatically for blocks containing error definitions. Diagnostics functions can be enabled for blocks which do not contain an error definition, but rather initial logic operations for the network to be monitored:

1. Activate the "Save Process Diagnostics Data" check box in the LAD/STL/FBD Editor, as described below.
2. Enable diagnostic functions for the block to which you want to append the error message as follows.

Enabling the Check Box

Procedure:

1. Doubleclick the selected block in SIMATIC Manager to open the LAD/STL/FBD Editor and call menu command **Options > Customize** .
2. Select the "Create Block" tab in the dialog box and set the "Store Process Diagnostics Data" check box. Exit the dialog box with "OK".

Enabling Diagnostics Function for Blocks

You have two means of enabling diagnostics functions for blocks:

- If you append an error definition to a block, it will be automatically assigned the diagnostics attribute.
- You can also enable a block to contain diagnostic data by assigning the following system attribute:

Attribute	Value	Assign this attribute if...	For Block
S7_pdiag	true	data are to be generated for S7-PDIAG.	FB, FC, OB, and DB

How to assign this system attribute to a block:

1. Open the block and select menu command **File > Properties** in the incremental LAD/STL/FBD Editor**.**
2. The dialog box containing the block properties is displayed.
3. Click the "Attributes" tab and enter the attribute specified in the table above.
4. Exit the dialog box with **"OK"** and save the block in the Editor via the menu command **File > Save**.
5. You have now enabled the diagnostic attribute for your block.

4 S7-PDIAG Monitoring Types

4.1 Address Monitoring

Address monitoring is directly linked to an address, namely the initial diagnostic address.

Monitoring Definition

The address monitoring function monitors whether the initial diagnostic address has acquired a specific level after a certain period of time (monitoring time). If the result is positive, the error is registered as an incoming error. The error is outgoing after the address changes its level again.

Depending on whether you select level or edge monitoring, the delay time is triggered either immediately or not until the next edge transition.

There are two address monitoring modes:

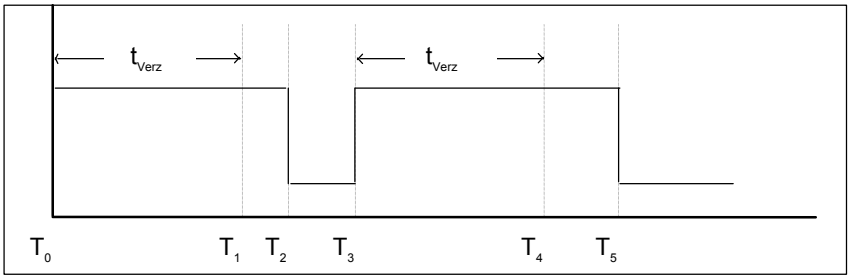
- Level monitoring and
- Edge monitoring:

Level Monitoring

The level monitoring function monitors a defined level (0 or 1) at the initial diagnostic address. An error state occurs if the address has maintained a specified level for a certain period of time. The user can define this delay time. A level change within this monitoring time retriggers the delay time.

Monitoring starts at the first cycle (**T**o = startup). The delay time (**tdelay**) starts both at T**0** and T3. If the length of the defined level signal exceeds the specified delay time, an ”incoming" error will be reported at **T**1 and T4. **T**2 and T5 report "outgoing" errors.

The figure below shows address monitoring in level monitoring mode.

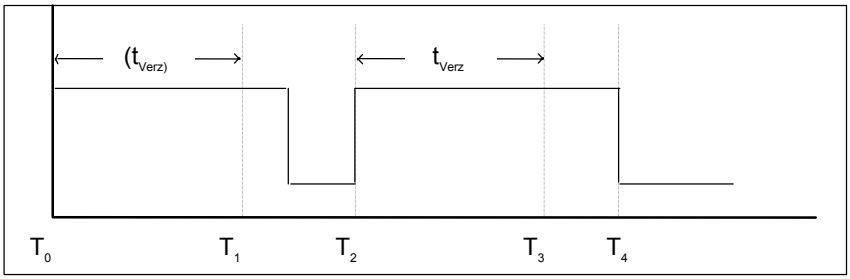


Edge Monitoring

The edge monitoring function is used to detect defined positive or negative edges ("0 >1" or "1 >0" transitions) at the initial diagnostic address. An error state occurs if the address maintains an incorrect level during the specified delay time after the edge transition (for example, level "1" after a positive edge). A level transition within this monitoring time retriggers the delay time.

Edge and level monitoring are basically the same functions, with the exception that edge monitoring stores the status of the monitored address at the time **To** (= startup), i.e. the startup point is not interpreted as an edge.

The next (selected) edge transition triggers the delay time (**tdelay**). An error will thus not be detected and reported "incoming" until the time **T3** starts, rather than at the time **T1** as would be the case when operating in level monitoring mode. The error is reported "outgoing" at the time **T4**.



Error Status

The address monitoring function will return an error status if the following user-defined conditions have been met.

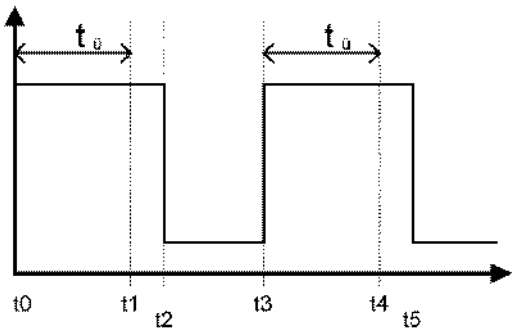
Assigning Messages

You should assign messages to your level and edge monitoring definitions. The operator panel displays these messages after an error has occurred.

Difference Between Level and Edge Monitoring

The following diagram shows the difference between level and edge monitoring. Level monitoring is used to detect errors at the times t1 and t4 and to report "outgoing" errors at the times t2 and t5.

Edge monitoring detects an incoming error at the time t4 and reports it as outgoing at the time t5.



4.2 Motions Monitoring

4.2.1 Overview of Motion Monitoring

S7-PDIAG provides four default monitoring functions, designed especially for monitoring motions within your process. Prerequisite is that you are using UDT_Motion and the LAD networks for motion monitoring.

Motion monitoring definitions have a preset logic you merely need to complement. When you declare a motion monitoring definition, refer to a predefined monitoring logic and expand and modify it accordingly. An error state occurs if the defined conditions are met.

You can select any of the following default motion monitoring definitions:

- S7-PDIAG: Action monitoring definitions
  monitor whether or not a motion has been completed within the specified action time. This is the case if the target final position was reached.
- S7-PDIAG: Startup monitoring definitions monitor whether or not a motion is actually started if all conditions have been met. This is the case if the object moves out of the current final position within the preset startup time.
- S7-PDIAG: Reaction monitoring definitions
  monitor whether or not the target position is maintained in a stable state without a reverse direction control command, or if the object drifts off this position for a time no longer than the specified reaction time.
- S7-PDIAG: Interlock monitoring definitions monitor whether or not the conditions required for a motion interlock have been met.

You should assign a message to your motion monitoring definitions. This operator panel displays this message after an error has occurred.

Notes

- Make sure to include the LAD networks for motion programming and UDT_Motion when programming motions in your user program. Maximum benefits are therefore achieved by using predefined motion monitoring logic for error monitoring.
- When you set the monitoring time to "0", it will be ignored and you therefore generate less code. This also implies that you can no longer use the Modify Times Online/Offline menu command.

4.2.2 Action Monitoring

Action monitoring represents one of the four motion monitoring modes. In this monitoring mode, the function checks whether the target final position (Target final position) has been reached within a certain time (Action time) after a control command was output (Trigger).

- The point at which the process is monitored is the target final position, i.e. the complete run sequence of a motion is monitored during the process.
- The monitoring logic is already defined. You only have to add the trigger and the action time.
- An error state occurs if the trigger for the specified action time has been activated and the target final position (initial diagnostic address) has not been reached.

Action monitoring is based on the <target final position>.

The action monitoring logic is defined as follows:

ONDT (<Trigger>,<Action_Time>**)**
AND
NOT <Target_Final_Position>

If you take advantage of the motion programming options of S7-PDIAG and used UDT_Motion in your program, the Initial Diagnostic Address will represent the "Final_Position[n]" for action monitoring.

In this case, the default monitoring logic is:

ONDT (Motion_Name.Control1/2, <Action_Time>**)**
AND
NOT Motion_Name.Final_Position[n]

"MotionName" is the name of the UDT_Motion, "Control1/2" is the name of the trigger. You only have to add the required monitoring time.

Notes

- Make sure that you include the LAD networks for motion programming and UDT_Motion when programming motions in your user program. Maximum benefits are achieved by using the predefined motion monitoring logic for error monitoring.
- When you set the monitoring time to "0", it will be ignored and you therefore generate less code.
- In error definitions, times can now be modified by means of a variable of data type TIME in addition to being entered a fixed values.

4.2.3 Startup Monitoring

Startup monitoring is one of the four motion monitoring modes. It is used to check whether an object moves out of the actual final position (Actual final position) within a certain time (Startup time) after a control command (Trigger) was output.

- The point at which the process is monitored is the current final position. The reaction of a motion to a control command is therefore being monitored. This means that this function detects an error at an earlier stage than the action monitoring function. This is particularly useful when operating with slow processes.
- The monitoring logic is predefined. You merely have to add the trigger and startup time parameters.
- The error state occurs if specified startup time has been triggered and the actual final position (initial diagnostic address) signal is still active.

Startup monitoring is based on the <actual final position> as initial diagnostic address.

The startup monitoring logic is defined as follows:

ONDT (<Trigger>,<Startup_Time>**)**
AND
<Actual_Final_Position>

If you take advantage of the motion programming options of S7-PDIAG and used UDT_Motion in your program, the Initial Diagnostic Address will represent the "Final_Position[n]" for startup monitoring.

In this case, the monitoring logic is preset as follows:

ONDT (Motion_Name.Control1/2, <Startup_Time>**)**
AND
Final_Position[n]

"MotionName" is the name of the UDT_Motion, "Control1/2" is the name of the triggering UDT variable Control 1 or Control 2. You only have to add the required monitoring time.

Notes

- Make sure to include the Ladder networks for motion programming and UDT_Motion when programming motions in your user program. Maximum benefits are achieved by using the predefined motion monitoring logic for error monitoring.
- When you set the monitoring time to "0", it will be ignored and you therefore generate less code.
- In error definitions, times can now be modified by means of a variable of data type TIME in addition to being entered a fixed values.

4.2.4 Reaction Monitoring

Reaction monitoring is one of the four motion monitoring modes. It monitors whether a final position which has been reached remains stable after a certain period of time has expired (reaction time).

- The monitoring point is the target final position. In order to monitor the target final position, you require an additional position flag, which can be found immediately before the actual final position in the process. The reaction of a motion to a triggering signal is therefore being monitored.
- The monitoring logic is predefined. You merely have to add the position flag and the reaction time parameters.
- The error state occurs if the position flag is active and the target final position (initial diagnostic address) cannot be reached within the specified reaction time, or if the target final position has been reached and the object moves out of this position longer than the specified reaction time.

Reaction monitoring is based on the <target final position> as initial diagnostic address.

The reaction monitoring logic is defined as follows:

ONDT (<Position_Flag> **AND NOT** <Target_Final_Position>,<Reaction_Time>)

If you take advantage of the options available for motion programming in S7-PDIAG and if you have used the UDT_Motion in your program, the initial diagnostic address for reaction monitoring represents the Final_Position[n]".

In this case, the monitoring logic is preset as follows:

ONDT (Motion_Name.Position_Flag[n] **AND NOT** Motion_Name.Final_Position[n],<Reaction_Time>)

"MotionName" is the name of the UDT_Motion. You merely have to add the required monitoring time.

Notes

- Make sure to include the Ladder networks for motion programming and UDT_Motion when programming motions in your user program. Maximum benefits are achieved by using the predefined motion monitoring logic for error monitoring.
- Input of times is optional for motion monitoring definitions. If you set the monitoring time of "0", it will be ignored and you therefore generate less code.
- In error definitions, times can now be modified by means of a variable of data type TIME in addition to being entered a fixed values.

4.2.5 Interlock Monitoring

Interlock monitoring is one of the four motion monitoring modes. It checks whether the interlock condition (Executability) has been met after a motion control command (Trigger) was output and a certain time (Interlock time) has expired.

- The point at which the process is monitored is the executability. This monitoring function checks whether the interlock condition is met after the motion has been triggered and the interlock time has expired.
- The monitoring logic is predefined. You merely have to add the trigger and the interlock time.
- An error state occurs if the specified interlock time has been triggered and the interlock condition (initial diagnostic address) is still inactive.

Interlock monitoring is based on the <executability> as initial diagnostic address.

The interlock monitoring logic is defined as follows:

ONDT (<Trigger>,<Interlock_Time>
AND
NOT<Executability>

If you take advantage of the options available for motion programming in S7-PDIAG and if you have used the UDT_Motion in your program, the initial diagnostic address for interlock monitoring represents "Executability1/2"

In this case, the monitoring logic is defined as follows:

ONDT (Motion_Name.Trigger1/2,<Interlock_Time>)
AND
NOT Motion_Name.Executability1/2

"MotionName" is the name of the UDT_Motion. You merely have to add the required monitoring time.

Notes

- Make sure to include the Ladder networks for motion programming and UDT_Motion when programming motions in your user program. Maximum benefits are achieved by using the predefined motion monitoring logic for error monitoring.
- Entering a time in motion monitoring definitions is optional. If you set the monitoring time to "0", it will be ignored and you therefore generate less code.
- In error definitions, times can now be modified by means of a variable of data type TIME in addition to being entered a fixed values.

4.2.6 Monitoring Times

S7-PDIAG provides the following monitoring times which you can declare via the dialog boxes as required for your error or monitoring definitions :

- Time in an address monitoring definition.
- Times in the S7-PDIAG language elements for the global monitoring definition.

Motion monitoring times:

- Action time
- Startup time
- Reaction time and
- Interlock time

You can also set a "0" time for the monitoring functions in order to save memory space at the AS.

Note

- If you do not enter a time or a time equal to "0", this implies automatically that these monitoring functions are excluded from "Online modification". You cannot modify a monitoring time you have set to "0" via the "Change Times online" function.
- In error definitions, times can now be modified by means of a variable of data type TIME in addition to being entered a fixed values.

4.2.7 Example of a Motion Monitoring Task

Processes often involve controlling of procedures which have two stable final positions and their transfer from one final position move to the other.

For example, a cylinder can be moved from its current final position to the target final position after switching on the hydraulic pressure. In the same way, a reactor can be heated up from a current temperature to a higher value once a heating system is switched on. These procedures can be interpreted as motions.

A motion thus represents a process sequence that is assigned the following properties:

- An object is in a current, stable final position
- A trigger is executed
- The object approaches and is able to reach the final target position due to an output control command.

Error Status

The motion monitoring function will return an error status if the following user-defined conditions have been met.

Example

The following Figure gives you an overview of how and where the various monitoring modes start to monitor the motion.

Default conditions:

- There is a motion with two final positions, the current final position (CFP) and the target final position (TFP).
- Each final position is assigned a position flag (PF) and a safety guard (SG) as an interlock enable.
- The control signal is to trigger a motion from the current final position (CFP) to the target final position (TFP). However, this motion may not start until the safety guard is closed.

Motion Monitoring Sequence:

- The control signal will be set at the time T0. Interlock monitoring starts at this point. If the interlock condition has been met and the function has recognized that all protective covers are closed within this time, the motion control trigger will be output (**T**1).
- Motion startup monitoring checks to see whether the object leaves the actual final position (CFP) within the startup time (T2). Monitoring starts once the interlock condition has been enabled (T1).
- The actual motion is monitored by the action time. After the interlock condition is enabled (**T**1), the function checks whether the object reaches the final target position (**T**4) within the action time. The completion of the motion is detected when the object reaches the final target position (T4).
- Reaction monitoring checks whether the target final position remains stable. The reaction time starts once the position flag (T3) is set or the object leaves the final target position while the position flag is still set.

4.3 Global Monitoring

The global monitoring function provides options for creating a tailor-made monitoring logic and performing complex error monitoring functions using S7-PDIAG language elements.

Monitoring Logic

The global monitoring lets specify a user-specific monitoring logic in the form of a logical expression sequence. Using the language elements provided in S7-PDIAG, the user can create a monitoring logic for complex error monitoring functions. An error state is detected if the defined conditions are met (if logically TRUE).

The Initial Diagnostic Address merely forms the initial address for criteria analysis. If this address is to be part of the monitoring definition (i.e.if it is to trigger the error), you must specify this explicitly.

Error Status

The global monitoring function also returns an error status if user-defined conditions have been met.

The following generally applies when detecting an error with a defined monitoring logic:

- The logical result "0" indicates that no error has been detected.
- The logical result ”1" indicates that an error has just been detected.
- An "incoming" error message will always be generated if the result is a logical "0" to "1" transition.
- An "outgoing" error message will always be generated if the result is a logical "1" to "0" transition.

Example of Global Monitoring

In the following example, the monitoring function determines whether all three safety guards of a press are closed, for example.

The following conditions are given:

- Protective cover 1: I1.0 = status 0 = Protective cover open.
- Protective cover 2: I3.5 = status 0 = Protective cover open.
- Protective cover 3: I7.2 = status 0 = Protective cover open.
- Controls: I5.0 = status 1 = Die stamp down.
- **Your monitoring logic will then look like this:**
- I5.0 AND NOT (I1.0 AND I3.5 AND I7.2)
- **Result:** An error event will be triggered if one of the protective covers is open.

Assigning Messages

You should assign an error message in your error definitions that is displayed on the logged on display devices after an error has occurred, i.e. if the monitoring logic is true.

5 Language Description for Programming Global Monitoring

5.1 Language Elements of S7-PDIAG

Using language elements, you can program your specific global monitoring logic.

It is important that you only use language elements which belong to the programming languages of S7-PDIAG when entering monitoring logic, and that you arrange these language elements in the correct syntax.

Any syntax errors in the monitoring logic are reported when you attempt to save your changes. All characters which are valid as identifiers for addresses or timers in STEP 7 are permitted.

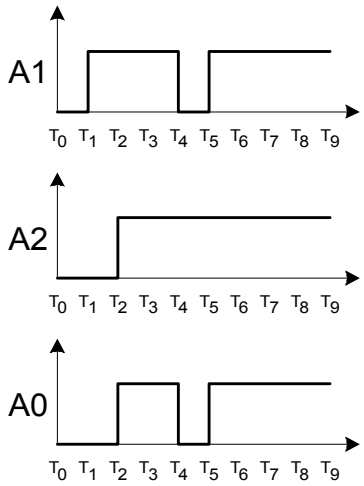
S7-PDIAG supports the following programming language elements:

- AND
- ONDT
- EN
- EP
- NOT
- OR
- SRT
- XOR
- Separators
- Brackets
- Addresses
- Timers
- Set and Reset Assignments

5.2 AND

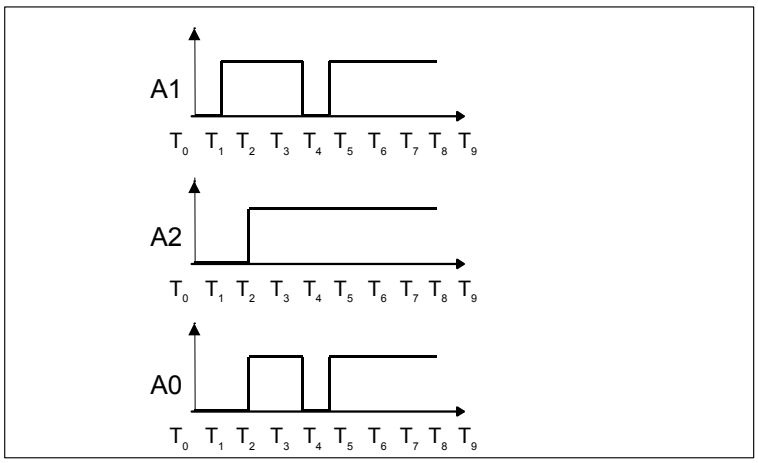
Syntax: A1 AND A2

The logic AND operation links the two expressions A1 and A2 to form a new expression A0, i.e. the expression A0 = TRUE if A1 AND A2 = TRUE. The result is also an expression and can be used for further logic operations.



Examples:

- I1.0 AND I1.1
- "MotorOn" AND "Enable"
- ("MotorOn" AND "Enable") AND "Automatic"



Expression A1 and A2 at AND

Note

If several expressions are logically linked using an AND operation, these expressions are still processed, even if one of the expressions is FALSE and the results of the expression is FALSE.

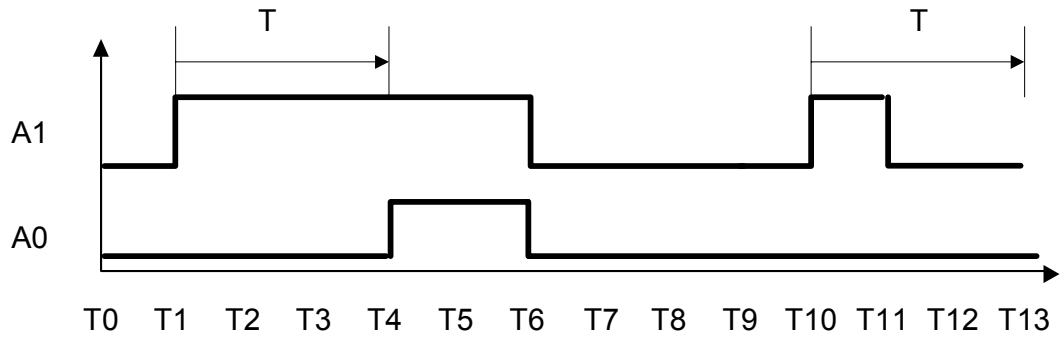
5.3 ONDT

Syntax: ONDT (A1, T)

The ONDT instruction executes an on delay. Depending on the expression A1 and the time T, the ONDT operation forms the expression A0 according to the following syntax:

- If (A1 = FALSE), then A0 = FALSE.
- If (A1 = TRUE) and (T running), then A0 = FALSE.
- If (A1 = TRUE) and (T expired), then A0 = TRUE.
- If A1 changes from FALSE to TRUE (positive edge), the timer will be retriggered (regardless of whether it is already running).
- If (A1 = TRUE), the timer starts (program start / complete restart).

The ONDT instruction always retriggers the delay time if the result of a logic operation is TRUE. If the time has expired, the result of ONDT is TRUE and can thus be used again.



Examples:

- ONDT ("Trigger", 1000)
  If "Trigger" is TRUE, the result of this expression is TRUE one second later, if "Trigger" remains TRUE.
- ONDT (I1.0, T#2s)
  When I1.0 = TRUE, the result = TRUE after two seconds, provided I1.0 remains TRUE.

ONDT actually represents a positive level monitoring function which can logically linked in further operations.

Note

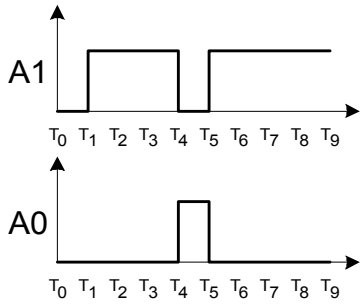
- The expression A1 should not contain any of the following qualifiers:
- ONDT
- EP
- EN
- SRT

5.4 EN

Syntax: EN (A1)

The EN instruction (negative edge) saves the result of the last (positive or negative) edge at expression A1. The EN instruction forms the expression A0 from the expression A1 according to the following syntax:

- A0 = TRUE after a negative edge (transition TRUE to FALSE) at A1.
- A0 = FALSE after a positive edge (transition FALSE to TRUE) at A1.
- A0 = FALSE until the occurrence of the first negative edge at A1.



EN thus evaluates whether the logic result of the input expression has caused a TRUE to FALSE transition. The result is TRUE when the TRUE to FALSE transition is detected. The result remains TRUE until the input expression is TRUE again.

Examples:

- EN(I1.0)
- EN("Trigger")

This is useful if you do not want to detect an edge transition until the monitoring definition has started, and thus skip the initial state. In this case the value of the input expression is evaluated on startup. If the input expression is already FALSE at this point, it is ignored.

Note

The expression A1 should not contain any of the following qualifiers:

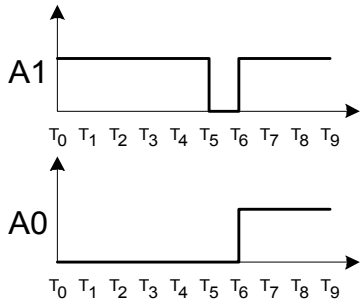
- ONDT
- EN
- EP
- SRT

5.5 EP

Syntax: EP (A1)

The EP instruction (positive edge) saves the result of the last (positive or negative) edge at expression A1. The EP instruction forms the expression A0 from the expression A1 according to the following syntax:

- A0 = TRUE after a positive edge (FALSE to TRUE transition) at A1.
- A0 = FALSE after a negative edge (TRUE to FALSE transition) at A1.
- A0 = FALSE until the occurrence of the first positive edge at A1.



EP thus evaluates whether the logic result of the input expression has triggered a FALSE to TRUE transition. The result is TRUE when the FALSE to TRUE transition is recognized. The result remains TRUE until the input expression is FALSE again.

Examples:

- EP(I1.0)
- EP("Trigger")

This is useful if you do not want to detect an edge change until the monitoring definition has started and thus skip the initial state. In this case the value of the input expression is evaluated on startup. If the input expression is already TRUE at this point, it is ignored.

Note

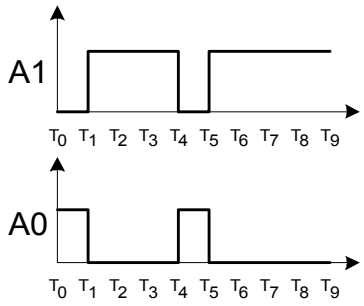
The expression A1 should not contain any of the following qualifiers:

- ONDT
- EN
- EP
- SRT

5.6 NOT

Syntax: NOT A1

The NOT instruction forms the expression A0 by inverting the expression A1. If A1 = TRUE, then A0 = FALSE. If A1 = FALSE, then A0 = TRUE. The NOT instruction inverts the logic result of the operation.



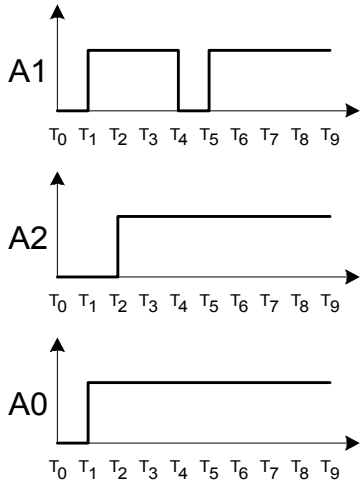
Examples:

- NOT I1.0
- NOT("Trigger1" AND "Trigger2")

5.7 OR

Syntax: A1 OR A2

The OR instruction is a logical link of two expressions A1 and A2 to form a new expression A0, i.e. the expression A0 = TRUE if at least one of the two expressions A1 OR A2 is TRUE. The result is also an expression and can be used in further logical operations.



Examples:

- I1.0 OR I1.1
- "MotorOn" OR "Enable"
- ("MotorOn" OR "Enable") AND "Automatic"

Note

If several expressions are logically linked by an OR instruction, these expressions are still processed, even if one of the expressions is TRUE and the result of the operation is therefore TRUE.

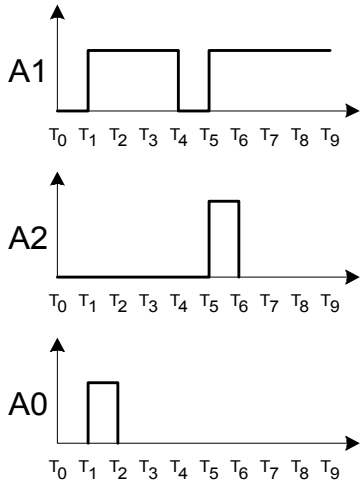
5.8 SRT

Syntax: SRT (A1, A2, T)

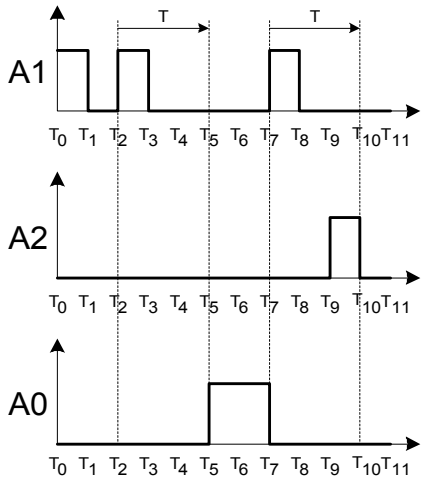
The SRT instruction (Set/Reset Timer) is an on delay unit, operated with one set and one reset input. The set/reset inputs are pulse-driven. This means that the event is triggered by a positive edge (pulse) and not by TRUE or FALSE. The SRT instruction forms the expression A0, depending on the expressions A1 (set input), A2 (reset input), and the time T according the following syntax:

- A positive edge (change from FALSE to TRUE) at A1 (set input) starts the time T, regardless of whether the timer is running. A0 is set to FALSE.
- A positive edge (change from FALSE to TRUE) at A2 (reset input) stops the time T, regardless of whether the timer is running. A0 is set to FALSE.
- A0 = TRUE if the time T has expired.
- A0 is initialized with FALSE on program start.
- If a simultaneous positive edge occurs at A1 (set input) and at A2 (reset input), A1 is ignored.

Special case: SRT if the time = "0":



SRT in general:



The SRT instruction monitors the status between two expressions. A positive transition at expression 1 triggers the delay time. A positive transition at expression 2 stops the delay time. If the delay time expires before it is stopped by a transition at expression 2, the logical status of the resultant expression is set to "1". The resultant expression maintains the same value until the next positive transition at expression 1 or expression 2. If the logical level of both expressions changes within the same cycle, the result is FALSE since expression 2 is of higher priority.

Examples:

- SRT(I1.0, I1.1, 1000)
- SRT("Trigger1", "Trigger2", 2000)

Note

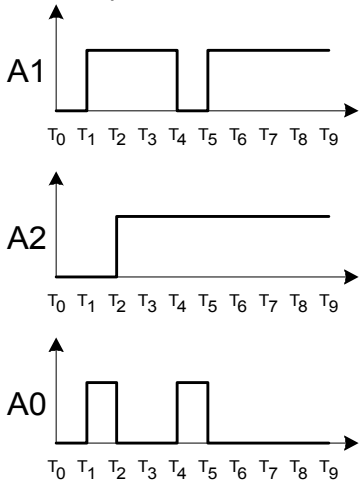
The expressions A1 and A2 should not contain any of the following qualifiers:

- ONDT
- EN
- EP
- SRT

5.9 XOR

Syntax: A1 XOR A2

The XOR instruction logically links two expressions A1 and A2 to form a new expression A0, the expression A0 = TRUE if A1 or A2 = TRUE. The result is also an expression and can be used for further logical operations.



Examples:

- I1.0 XOR I1.1
- "MotorOn" XOR "Enable"
- ("MotorOn" XOR "Enable") XOR "Automatic"

5.10 Separators

You must separate the individual language elements by using separators. If you put a separator in quotation marks ("..."), the character will not be evaluated as a separator.

The following characters are interpreted as separators:

- Space
- TAB
- RETURN/ENTER
- Brackets

5.11 Brackets

Brackets determine the processing sequence.
Brackets can also be used as separators.

5.12 Addresses

You can use all binary S7 addresses. Symbolic names must be enclosed in quotation marks.

5.13 Timers

The qualifiers ONDT and SRT supply a time. The unit of the time value is automatically interpreted in milliseconds (ms) if you enter a numeric value. You can enter a value either in S7-Time format (T#...) or in accordance with the IEC standard (7h5m6s). The resolution depends on the AS you are using.

In addition to the entry as a fixed value, timers can also be controlled via the variable data type TIME as of S7-PDIAG V5.1.

5.14 Set and Reset Assignments

If you intend to monitor a program sequence with a set and reset assignment in S7-PDIAG, as shown in the example below, you will receive a warning during generation that this is not permissible.

An expression from NOT and an assignment is not a valid expression from the S7-PDIAG point of view. Thus, an initial value acquisition is not possible for it.

Permissible Programming:

```
A I 0.0
R M 50.0
S M 50.1
NOT
= Q1.0 => IDA
```

Impermissible Programming:

```
A I 0.0
NOT
= Q1.0 => IDA
```

Or:

```
A I 0.0
NOT
R M 50.0
S M 50.1
= Q1.0 => IDA
```

5.15 Language Syntax

The language syntax describes the interrelation of language elements. The valid S7-PDIAG syntax for defining monitoring definitions comprises various expressions. An expression is a logic operation with binary addresses and can acquire the value TRUE or FALSE. The result of a logic operation then represents a binary result which can be interconnected. An expression can consist of a logic operation with several expressions or just one address. There are expressions and Boolean expressions.

This section illustrates the valid syntax for the following:

- Expressions and
- BOOLEAN expressions.

The words in bold text are metawords in the language.

Expressions are always binary and can acquire the value TRUE or FALSE.

Examples of Expressions:

- I1.0
  This is the simplest case of an expression, namely a single address.
- I1.0 AND I1.1
  The two addresses I1.0 and I1.1 are logically linked by AND.
- (I1.0 AND I1.1) OR (I1.2 XOR I1.3)
  The logic operations in brackets are executed first and then the two results are logically linked by OR.
- ONDT(NOT I1.0, 1000)
  This expression comprises a Boolean expression and a time.

The following section shows you the valid syntax for expressions and Boolean expressions. The syntax for addresses and timers corresponds to the standard syntax in STEP 7. You should adhere to this syntax when you enter your monitoring logic:

Syntax of expressions:

Expression:=	Address	*or*
	(expression**)**	*or*
	NOT expression	*or*
	expression **OR** expression	*or*
	expression **AND** expression	*or*
	expression **XOR** expression	*or*
	EP (Boolean expression**)**	*or*
	EN (Boolean expression**)**	*or*
	ONDT (Boolean expression, time**)**	*or*
	SRT (Boolean expression, Boolean expression, time**)**	

Syntax for Boolean Expressions:

Boolean Expression:=	Address	*or*
	(Boolean Expression**)**	*or*
	NOT Boolean Expression	*or*
	Boolean Expression **OR** Boolean Expression	*or*
	Boolean Expression **AND** Boolean Expression	*or*
	Boolean Expression **XOR** Boolean Expression	

Notes

- If the result of logic operation for the whole expression is TRUE, the error is registered as "incoming." If the result of the whole expression is FALSE, the error is registered as "outgoing."
- Note that the maximum number of addresses may not exceed 64.

5.16 Priority of Language Elements in the Monitoring Logic

The qualifiers are processed according to the following priority:

Qualifier	Priority
()	1
EP, EN, ONDT, SRT	2
NOT	3
AND	4
XOR	5
OR	6

If qualifiers have the same priority, the monitoring logic is processed from left to right.

Note

- If several expressions are logically linked by AND, these expressions are still processed, even if one of the expressions is FALSE and results in the whole expression is thus FALSE.
- If several expressions are logically linked by OR, these expressions are still processed, even if one of the expressions is TRUE and results in the whole expression being TRUE.

5.17 S7-PDIAG Checks

S7-PDIAG performs the following checks:

1. Language element check.
2. Address range check.

Note

Always use S7-PDIAG language elements and arrange them according to the proper syntax. Otherwise, an error will be displayed both when you enter the language elements and when you attempt to compile the monitoring logic.

6 Programming Support and Data Structure in S7-PDIAG

6.1 The Concept of Motion Programming in S7-PDIAG

The motion programming concept in S7-PDIAG is based on three components:

1. The included LAD networks for motion control programming and the corresponding data structure for the motion, namely the UDT_Motion (UDT 2),
2. The four special motion monitoring modes under S7-PDIAG and
3. The display devices which are tuned to the data structure of the UDT_Motion.

LAD Networks for Motion Programming

These networks required for motion programming under S7-PDIAG contain by default all interconnections required for motion control, e.g.:

- Check of interlock conditions
- Control of auto mode
- Control manual mode

These networks always describe one direction in the process and are therefore required twice for each motion.

In order to simplify programming when using these special networks, they are grouped in the motion control block FB100 found in the included example file "ZEn06_01_S7PDIAG_drill". These networks provide and clear data in the included UDT_Motion (UDT2) function that represents the interface to the display units.

UDT_Motion

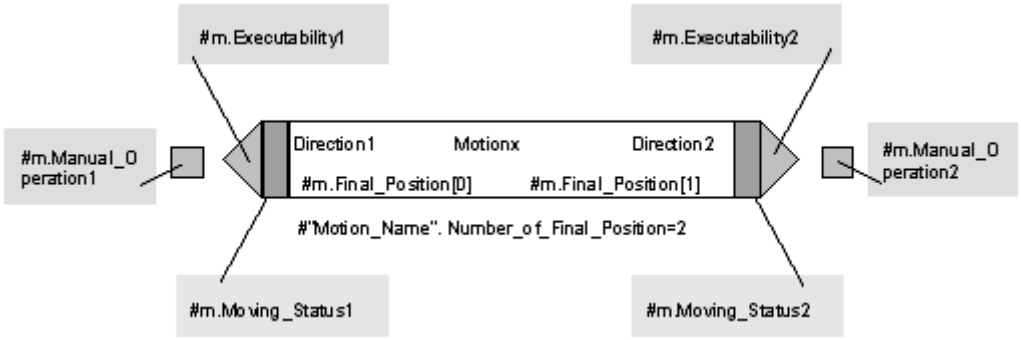
The data structure of the UDT_Motion forms the interface to the display units (HMI). The display devices recognize this data structure and can thus directly access specific data. "ZEn06_01_S7PDIAG_drill" automatically determines the reference.

The display devices can recognize whether or not a motion is currently active or not, based on the data provided. The operator panels can also be used to control motions by setting specific bits in manual mode, provided this operating mode has been selected. The example "ZEn06_01_S7PDIAG_drill" included uses the data structure of UDT_Motion in FB100.

Output of UDT_Motion Data Elements on the Display Unit (HMI)

Troubleshooting routines can be performed comfortably by means of hotkeys on the display unit, for example. Hotkeys are directly interconnect the digital control inputs to the digital outputs of the operator panel (e.g. wired or via DP interface) and thus allow instantaneous operator control of motions.

However, it is in this case required that you have used UDT "Motion" for motion programming and thus incorporated a standardized interface to the motion screen. Motions are displayed on the display unit in a semi-graphical form. The figure below shows a screenshot of UDT_Motion elements on the display unit.



Advantages of Motion Programming

The advantages of the concept of motion control programming under S7-PDIAG:

- A motion can be monitored with error definitions and automatically contains a group error definition.
- A motion can be visualized on the motion screens of the operator panel without the need for any additional configuration efforts.
- A motion can be visualized in all the operating modes of the machines and can also be controlled via the operator panel (depending on the control program).
- The position and the motion is visualized using up to 16 final positions and the motion status.
- The motion has two directions for which you can define the direction texts. The executability can be formed in the user program for each motion direction. You are thus shown which motion can be controlled at the current machine status.

6.2 Support for Programming Error Definitions

S7-PDIAG provides support for assigning error definitions to system units or motions, by means of:

- user data types (UDTs)
  - the UDT_Unit,
  - the UDT_S_Unit and
  - the UDT_Motion
- automatic use of auxiliary networks that support criteria analysis, if you have also programmed preceding logic operations.
- LAD networks for motion programming that describe the parameter assignments at UDT_Motion whose data elements can be output to the operator panel (HMI).

What is a UDT?

A UDT is a userdefined data type which can be saved as a block. This means that you can create one UDT and then use it many times:

- as "standard" data type on the one hand,
- as template for generating blocks of the same data structure on the other.

Advantage of UDT: Bit ID for group errors

All UDTs are assigned one bit for group error detection which S7-PDIAG sets after an error has occurred. S7-PDIAG in this case set this bit in all higher-level units and motions.

Example of Using the Group Error ID

A punch press contains the elements "Press," "Safety Guard," and "Punch," where the safety guard and the punch are, in turn, components of the press.

The press is a unit in the sense of S7-PDIAG. The punch represents a motion which is enabled when the safety guards are closed.

In order to co-ordinate those objects, you need to program a "Coordination" FB that calls UDT_Unit and UDT_Motion.

When your display unit outputs a group error message informing you of a disruption at the "Press" unit, you can position the cursor in the overview screen on the "Press" unit and enter the next lower level, where you can view the three motion structures of the press, stamp and safety guard. This information also shows that a group error bit is set in the motion structure of the stamp.

Result: There is a fault on the press due to a stamping error. The fault in the punch has caused the error definition.

Benefits: The use of UDT_Unit, UDT_S_Unit and UDT_Motion allows you to locate errors by means of data reduction and to eliminate errors in manual mode.

Using Auxiliary Networks

S7-PDIAG can substitute addresses (bit memories and outputs if selected) in your program logic with your actual networks and then uses these for criteria analysis.

Example of the use of auxiliary networks

You have used the following logic operation in an FB that is assigned diagnostic functions:

- I1.0 AND I1.1 = M1.0

The following appears in the diagnostics data of another block:

- I1.2 AND M1.0 = Q1.1

After insertion of the auxiliary network, the results is:

- Q1.1 = (I1.2 AND (I1.0 AND I1.1))

When using auxiliary networks, brackets are always used to resolve the logic operation originally carried out with a memory bit.

Note

Please ensure that you have set this option under "**Options > Customize**".

LAD Networks for Motion Programming

S7-PDIAG provides LAD networks in an FB100 program of the included project "ZEn06_01_S7PDIAG_drill" in order to support motion programming:

- For motions in one direction using the hotkeys of the operator panel, and
- For motions in one direction using neither the hotkeys, nor manual operation, nor the status displays on the operator panel.

Using these networks for reduces motion programming efforts down to the simple entry of parameters you are using in the framed subnets.

These networks offer a convenient method of programming motions as well as the advantage of actually enabling operator control and troubleshooting of motion processes on the motion screen of the operator panel, for example.

Data Definition in the Examples

In these examples, we shall assume the existence of the following definition of a cylinder motion named "z" as well as of two further bits for setting auto and manual mode for the machine.

The data structure looks like this:

- z UDT_Motion
- auto BOOL
- manualBOOL

Note

The "auto" and manual" variables are usually fetched from the corresponding UDT_Unit. To keep matters as simple as possible, we have ignored this feature in our examples.

6.3 Definition of the User FB Interface (FB46)

The sample program in the project "ZEn06_01_S7PDIAG_drill" supplied with S7-PDIAG contains a user FB (FB 46) with interface definition.

The user FB allows you to direct your reaction towards specific error messages and thus to error states, without further programming effort.

This block is always called when an incoming or outgoing error is detected. You thus obtain, for example information on the unit, the priority and the message ID of the error.

Interface Definition of the User FB:

VERSION : 0.0

VAR_INPUT		// Standardized interface
	EV_C: BOOL;	// Incoming message if TRUE
		// Outgoing message if FALSE
	EV_ID : DWORD;	// Error message ID
	SD_1: ANY	// Associated value
	PRIO: BYTE	// Priority of the error message
	EV_DB: WORD	// DB number of the reporting unit
	USER_OPD: ANY	// specific address V5.0: NIL
END_VAR		// Standardized interface
VAR		// Free parameter
...		
END_VAR		// Free parameter
VAR_TEMP		// Free parameter
...		
END_VAR		// Free parameter
BEGIN		// Any instructions for processing the input parameters
END_FUNCTION_BLOCK		

Notes

- The associated value SD_1 represents the value specified in the error definition.
- The USER_OPD parameter is currently not used.

6.4 Support of the Instance/Type Concept

S7-PDIAG supports the type/instance concept of SIMATIC S7.

This means the user can program all error definitions directly at the corresponding block type, i.e. at the function block (FB).

S7-PDIAG will then generate the instances for error definitions, including the various corresponding messages, analogous to the instance DBs in your user program.

Generating Instance-Specific Message Texts

You can replace formal addresses by the name of the unit or motion in message texts.

If necessary, the final position names of motions can be automatically preset with the symbolic names configured in the block.

6.5 Using Auxiliary Networks

The auxiliary networks generated by S7-PDIAG are networks which describe the initial logic operations you are using. This initial logic operation is used again in a network which that is to be analyzed. S7-PDIAG uses auxiliary networks for criteria analysis.

The use of auxiliary networks will be aborted at the point where the system detects an ambiguous address. Criteria analysis is not possible in this case.

Run Sequence

The system ignores the run sequence of the various networks when using auxiliary networks.

Example:

```
A M0.0
A I1.0
= M1.1      //M1.1 has the value of cycle n-1

A I1.1
= M 0.0     //M0.0 has the value of cycle n
```

result of the insertion of the auxiliary network:

```
A I1.1
A I1.0
= M1.1      //only cycle n is taken into account.
```

Canceling the Auxiliary Networks

The use of auxiliary networks is canceled under the following conditions:

- The are located in blocks that are not capable of diagnostics; that is, they do not have the attribute S7_pdiag = true.
- The addresses to be replaced are themselves defined as an initial diagnostic address (IDA)
- An address to be replaced is defined as a cancel address.

6.6 Using LAD Networks for Motion Programming (FB 100)

6.6.1 Using LAD Networks for Motion Programming

Use the LAD networks for motion programming in FB100 of the project "ZEn06_01_S7PDIAG_drill" supplied with S7-PDIAG as well as UDT_Motion in order to avoid motion configuration errors.

If you use the UDT_Motion and the Ladder networks supplied for your motion programming, all the data required for a motion are generated for the display directly in the user program, without the need for further configuration.

Depending on the display device (HMI), you can use the hotkeys on the side of the operator panel for operator control of motions.

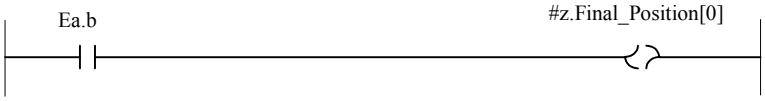
6.6.2 LAD Networks for Motion Programming

Complete Example for One Direction of a Motion Using Hotkeys

The following example shows how the system provides the incoming and outgoing values of the motion structure UDT_Motion. It is entirely up to you to decide on how you are going to utilize and edit the various networks. However, such actions may restrict the functions of display devices (HMI).

Network 1

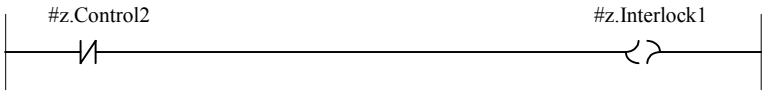
In network 1, the final position can be determined via a limit switch, a light barrier, or a combination of data. The status of the final positions is output on the motion screen, which also visualizes limit switch "Ea.b".



Network 1: Display final position [0]

Network 2

In this network, the system checks the safety conditions (interlocks) of the motions towards direction 1. In the example, the opposite direction of the motion is used; in this case, the trigger for the output is inverted.

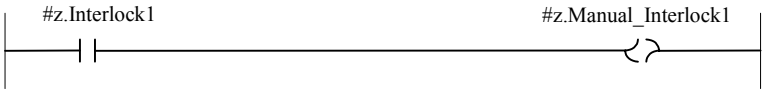


Network 2: Interlock direction 1

Network 3

In this network, the system checks the interlocks for operator control of motion towards direction 1.

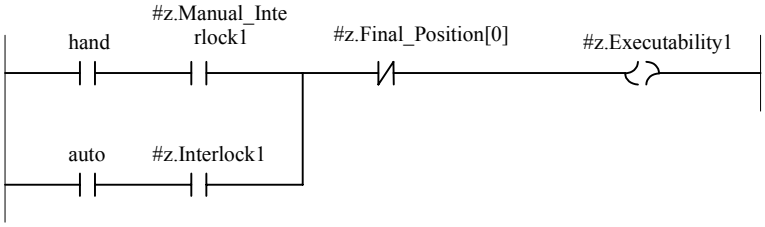
This network may or may not be used, depending on whether there are differences in the interlock between the operating modes. Depending on the application used, network 1 can also include the interlocks for automatic operation and network 2 the interlocks for operator control (programmed manual mode). In the example, the same interlocks apply as in network 1.



Network 3: Manual interlock direction 1

Network 4:

In this network, the executability is formed. This supports controlled manual mode. The network shows that motion is enabled towards direction 1.



Network 4: Executability direction 1

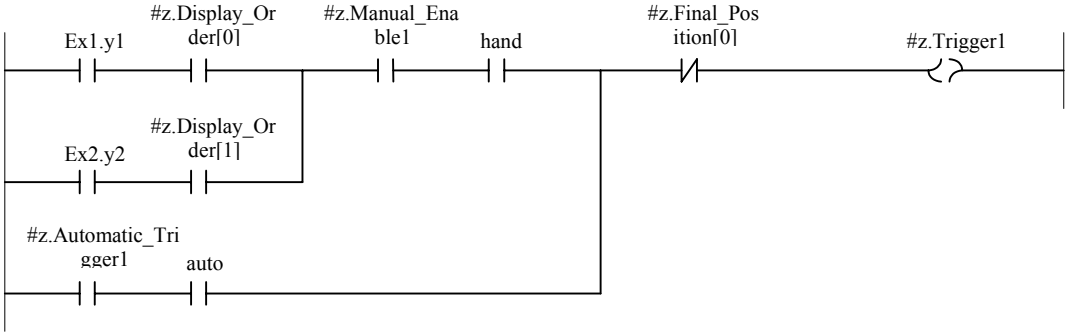
Network 5:

Here the trigger is formed which is to initiate the motion towards direction 1.

The lowest branch in this network represents auto mode. Due to the set process sequence, the variable #z.Automatic_Trigger1" is set at another location in the user program in order to initiate a motion.

The two upper branches in the network represent operator control of the motion via the motion screen, using the hotkeys. The #z.Manual_Enable1" parameter determines whether the motion is visualized on the display device or not.

"Ix1.y1" and "#z.Display_order[0]" would switch if the motion were first in the sequence and the hotkey Ix1.y1" were pressed. The same applies to all other hotkeys. In this example, only two hotkeys are used.



Network 5: Trigger direction 1

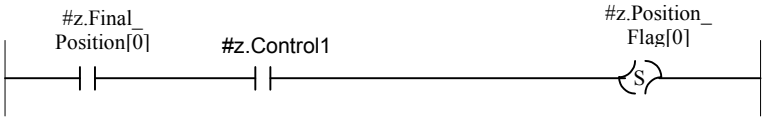
Network 6

This network is only required if you are using reaction monitoring.

The position flag is set when the motion has reached the final position and the trigger is still active. The trigger is then cleared in the next network. Reaction monitoring will be enabled as soon as the position flag is set.

Note

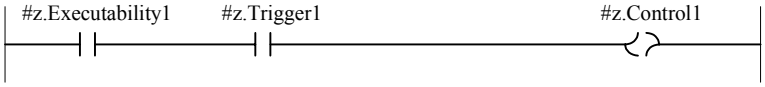
Reaction monitoring is enabled only if the motion has already been triggered once, in order to avoid initialization problems.



Network 6: Sets the position tag for reaction monitoring

Network 7:

In this network, the motion trigger is formed for all operating modes. In this example, the motion is only controlled if both the executability and the trigger for this direction are set.



Network 7: Triggers motion towards direction 1

Network 8:

The "Moving_Status" bit is used to indicate on the display unit whether the motion is actually executed or not.

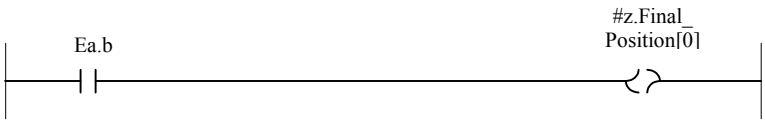
This can be determined implicitly by triggering the output, as shown in the example, or by measuring the motion directly in the process.

Network 8: Displays motion towards direction 1

Network 9

This network is only required if you are using reaction monitoring.

The position flag is reset when the motion is triggered in the opposite direction.



Network 9: Position flag reset

Shorter Example for One Direction of a Motion Without Using Hotkeys

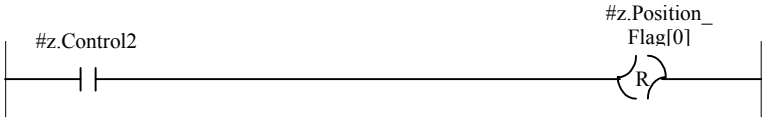
In the following example, one direction of a motion is shown in reduced form. This contains the display functions and monitoring definitions and excludes reaction monitoring

This motion cannot be operated via the hotkeys.

Network 1

The final position can be determined via a limit switch, a light barrier or a combination of data. The status of the final positions is displayed on the motion screen.

In this example, the limit switch Ia.b is visualized.



Network 1: Display final position [0]

Networks 2, 3 and 4

These networks form the conditions for execution and also contain the auto and manual mode interlock definitions. This supports controlled manual mode, and the networks show that motion is enabled towards direction 1.

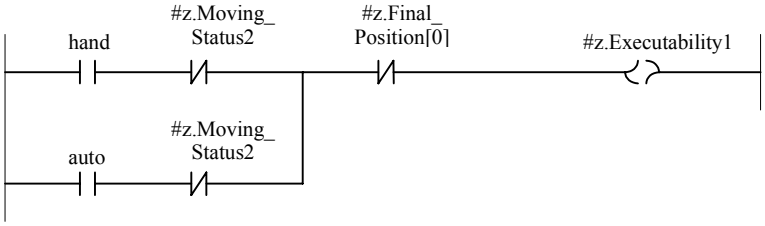
You can add further interlock conditions in these networks.

- **Interlock monitoring:**
  The interlock monitoring function is appended to the "Executability" signal. This functions, in combination with a criteria analysis, enables you to determine the missing signal which is preventing the motion.

  Monitoring logic (without time factor):
  #z.Trigger1 AND NOT #z.Executability1

  Monitoring logic (with time factor):
  ONDT (#z.Trigger1, ?) AND NOT #z.Executability1

  Monitoring logic (with time factor):
  ONDT (#z.Trigger1, ?) AND NOT #z.Executability1



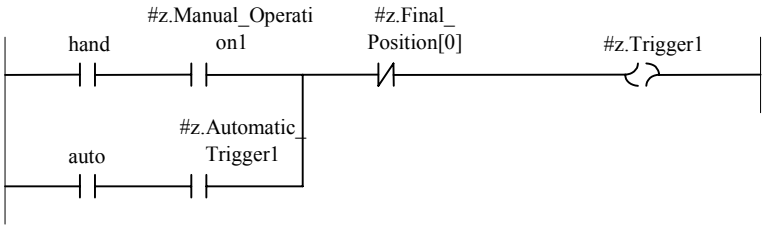
Network 2/ 3/ 4: Indicate executability of direction 1 on the display device (HMI)

Networks 5 and 6

Forms the trigger which is to initiate the motion towards direction 1.

The upper branch in the network shows key operated motions via the motion screen. When the key is pressed, the display device sets the "Manual_Operation" bit.

The lowest branch of this network represents auto mode motion: Due to the process sequence, the variable "#z.Automatic_Trigger1" will be set at another location of the control program.

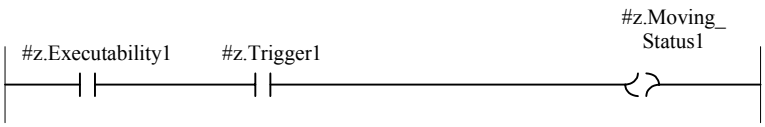


Network 5/6: Trigger direction 1

Networks 7 and 8

These networks form the motion trigger for all operating modes.

In this example, the motion is only controlled if both the executability and the trigger for this direction are set.



Network 7/ 8: Actuation/Display of motion towards direction 1

6.7 UDTs in S7-PDIAG

6.7.1 Overview of UDTs in S7-PDIAG

A UDT (User Data Type) is a userdefined data type which can be saved as a block. This means that you can create one UDT and then use it many times:

- as "standard" data type on the one hand,
- as template for generating blocks of the same data structure on the other.

The data structures for S7-PDIAG process diagnostics are defined by the following UDTs:

UDTs available in S7-PDIAG:

UDT_Unit:

The UDT_Unit comprises the information required in order for the operator panel (HMI) to assign an alarm message to the faulty program location. The UDT_Unit contains definitions for the following:

- Group error detection and group error acknowledgement
- for 16 modes, two of which are defined by default, namely the "Manual" and "Auto" modes. The user can define the remaining 14 operating modes as required.

UDT_S_Unit:

The UDT_S_Unit comprises the group error address and the group error acknowledgement.

This feature saves memory space and you do not need to "loop" the process unit mode definition to all subunits.

UDT_Motion:

The UDT_Motion represents a standardized interface between S7-PDIAG and the display devices (operator panels) and contains all parameters for:

- Displaying motions on the motion screens of the operator panels (HMI), without the need for additional configuration, and
- For manually moving these objects via the motion screen of the operator panel.

Note

Prerequisite is the use of LAD networks for motion programming, included in FB100 of the sample project "S7_DIAG".

The above UDTs are supplied with the S7-PDIAG software.

Group Error Bit ID

All UDTs contain a group error detection bit that is set by S7-PDIAG after an error has occurred. S7-PDIAG in this case sets this bit in all superimposed units and motions.

Example of an Application Using UDT_Unit and UDT_Motion

A stamping machine contains the elements ”Press," ”Safety Guard," and ”Punch," where the safety guard and the punch are, in turn, components of the press.

The press is a unit in the sense of S7-PDIAG. The punch is a motion which is enabled by the safety guard.

In order to co-ordinate those objects, you need to program a "Co-ordination" FB that calls UDT_Unit and UDT_Motion.

If you now receive a group error message on your display device informing you that the unit ”Press" is faulty, you can position the cursor in the motion screen on the unit ”Press" and move down one level to display the three motions for the press, punch, and safety guard. This information also shows that a group error bit is set in the motion structure of the stamp.

Result: There is a fault on the press due to an error in stamp operation. The fault in the punch has caused the error definition.

Benefits: The use of UDT_Unit, UDT_S_Unit and UDT_Motion allows you to locate errors by means of data reduction and to eliminate errors in manual mode.

6.7.2 UDT_Unit

The UDT_Unit contains the information required by the HMI to associate an error alarm with the faulty program location. The UDT_Unit contains defines:

- group error detection and group error acknowledgement
- 16 modes, with two default definitions of "Manual" and "Auto". You can define the remaining 14 operating modes to suit your requirements.

6.7.3 Using the UDT_Unit

Use UDT_Unit only once for each process unit (FB). All of the components subordinate to this process unit should refer to the operating mode of the higher-level unit.

Example: A press production line may contain many presses. They operate relatively independently and in different modes (manual, automatic). Each of the presses can be seen as a process unit, which contains further units (for example, the stamp, the protective cover, etc.). However, it is useful to control these subordinate units separately when they operate in different modes. For this reason, the subordinate units and their corresponding process units are set to the same operating mode.

6.7.4 Data Structure of the UDT_Unit

The UDT_Unit is characterized by the S7_pdiag_unit" attribute, and that this attribute is set "TRUE".

The table below shows the data structure of the UDT_Unit. You **may not** modify the default structure.

Address	Variable	Data Type	Initial Value	Comment
0.0		STRUCT		
+0.0	Unit_Version	WORD	B#16#1	Version of UDT (W#16#2 as of S7-PDIAG V5.0 HF3)
+2.0	Select_Automatic	BOOL	FALSE	1st operating mode selection
+2.1	Select_Manual	BOOL	FALSE	2nd operating mode selection
+2.2	Select_Operating_Mode2	BOOL	FALSE	3rd operating mode selection
+2.3	Select_Operating_Mode3	BOOL	FALSE	4th operating mode selection
+2.4	Select_Operating_Mode4	BOOL	FALSE	5th operating mode selection
+2.5	Select_Operating_Mode5	BOOL	FALSE	6th operating mode selection
+2.6	Select_Operating_Mode6	BOOL	FALSE	7th operating mode selection
+2.7	Select_Operating_Mode7	BOOL	FALSE	8th operating mode selection
+3.0	Select_Operating_Mode8	BOOL	FALSE	9th operating mode selection
+3.1	Select_Operating_Mode9	BOOL	FALSE	10th operating mode selection
+3.2	Select_Operating_Mode10	BOOL	FALSE	11th operating mode selection
+3.3	Select_Operating_Mode11	BOOL	FALSE	12th operating mode selection
+3.4	Select_Operating_Mode12	BOOL	FALSE	13th operating mode selection
+3.5	Select_Operating_Mode13	BOOL	FALSE	14th operating mode selection
+3.6	Select_Operating_Mode14	BOOL	FALSE	15th operating mode selection
+3.7	Select_Operating_Mode15	BOOL	FALSE	16th operating mode selection
+4.0	Automatic	BOOL	FALSE	1st operating mode of process unit
+4.1	Manual	BOOL	FALSE	2nd operating mode of process unit
+4.2	Operating_Mode2	BOOL	FALSE	3rd operating mode of process unit
+4.3	Operating_Mode3	BOOL	FALSE	4th operating mode of process unit
+4.4	Operating_Mode4	BOOL	FALSE	5th operating mode of process unit
+4.5	Operating_Mode5	BOOL	FALSE	6th operating mode of process unit
+4.6	Operating_Mode6	BOOL	FALSE	7th operating mode of process unit
+4.7	Operating_Mode7	BOOL	FALSE	8th operating mode of process unit
+5.0	Operating_Mode8	BOOL	FALSE	9th operating mode of process unit
+5.1	Operating_Mode9	BOOL	FALSE	10th operating mode of process unit

Address	Variable	Data Type	Initial Value	Comment
+5.2	Operating_Mode10	BOOL	FALSE	11th operating mode of process unit
+5.3	Operating_Mode11	BOOL	FALSE	12th operating mode of process unit
+5.4	Operating_Mode12	BOOL	FALSE	13th operating mode of process unit
+5.5	Operating_Mode13	BOOL	FALSE	14th operating mode of process unit
+5.6	Operating_Mode14	BOOL	FALSE	15th operating mode of process unit
+5.7	Operating_Mode15	BOOL	FALSE	16th operating mode of process unit
+6.0	Group_Error	BOOL	FALSE	TRUE = Unit failed
+6.1	Confirm_Units	BOOL	FALSE	TRUE: Unit will be acknowledged. Set by the display device (if configured) when the unit is acknowledged by user. Bit must be reset by user program.
+8.0	HMI_ID	DWORD	DW#16#0	Assigning the unit via display device (as of S7-PDIAG V5.0 HF3).
=12.0		END_STRUCT		

6.7.5 UDT_S_Unit

UDT_S_Unit only contains the group error address and the group error acknowledgement.

This saves memory space, and the process unit mode definition does not need to be "looped through" to all subunits.

6.7.6 Using the UDT_S_Unit

Use UDT_S_Unit only once in each process unit (FB).

The UDT_S_UNIT only contains the group error address and the group error acknowledgement. This means that you can structure your user program in such a way that it does not contain an operating mode definition. This save memory space and you do not need to "loop" the process unit mode definition to all subunits.

6.7.7 Data Structure of the UDT_S_Unit

The UDT_S is characterized by the "S7_pdiag_s" attribute, and this attribute is set "TRUE".

The following table shows the data structure of the UDT_S_Unit. You **may not** modify the default structure.

Address	Variable	Data Type	Initial Value	Comment
0.0		STRUCT		
+0.0	Unit_Version	WORD	B#16#1	
+2.0	Group_Error	BOOL	FALSE	TRUE = Unit failed
+2.1	Confirm_Units	BOOL	FALSE	TRUE: Unit will be acknowledged. Set by display device (if configured) when unit is acknowledged by user. Bit must be reset by user program.
= 4.0		END_STRUCT		

6.7.8 UDT_Motion

UDT_Motion represents a standardized interface between S7-PDIAG and the HMI. It contains all parameters required

- to display motions on the motion screens of the HMI, without the need of additional programming, and
- to control of those motions on the motion screen of the HMI.

Note

Prerequisite for motion programming is the use of the LAD Networks for Motion Programming in FB 100 of the sample project "ZEn06_01_S7PDIAG_drill" which is supplied in the S7-PDIAG software package.

6.7.9 Using the UDT_Motion

You should use the UDT_Motion each time you program a motion. The UDT_Motion represents the data interface between the currently active user program, S7-PDIAG, and the display devices.

The UDT_Motion contains all the parameters necessary for motion monitoring and control and can be inserted in the variable declaration table of a function block (FB):

- In the areas: "in", "out" and "stat".

If you use the UDT_Motion and the Ladder networks supplied for motion programming, all data required for a motion are generated for the display directly in the user program, without the need for further configuration.

6.7.10 Data Structure of the UDT_Motion

The UDT_Motion is characterized by the "S7_pdiag_motion" attribute, and that this attribute is set "TRUE."

The table below shows the data structure of the UDT_Motion. You **may not** modify the default structure.

Address	Variable	Data Type	Initial Value	Comment
0.0		STRUCT		
+0.0	M_Version	WORD	B#16#1	Version number of the UDTs (W#16#2 as of S7-PDIAG V5.0 HF3).
+2.0	Data_Length	BYTE	B#16#0	Length of motion structure (not currently evaluated).
+3.0	Moving_Status1	BOOL	FALSE	TRUE (bit, display device reading): object currently moving towards direction 1 (flashing rectangle).
+3.1	Moving_Status2	BOOL	FALSE	TRUE (bit, display device reading): the object is currently moving towards direction 2 (flashing rectangle).
+3.2	Executability1	BOOL	FALSE	TRUE (bit, display device reading): Motion towards direction 1 is locked due to Interlock 1, for example (filled arrow).
+3.3	Executability2	BOOL	FALSE	TRUE (bit, display device reading): Motion towards direction 2 is locked due to Interlock 2, for example (filled arrow).
+3.4	Group_Error	BOOL	FALSE	TRUE (bit, display device reading): a monitoring function whose IDA element represents an instance of this data structure has detected an error.

Address	Variable	Data Type	Initial Value	Comment
+4.0	Number_of_Final_Position	BYTE	B#16#0	Number of final positions used (byte, display device reading): The display device shows the actual number of final positions for this motion.
+6.0	Final_Position	ARRAY [0 to 15]	FALSE	TRUE (bit array, display device reading): one or several of max. 16 final positions have been reached (Final_Position_[0] is left).
*0.1		BOOL		
+8.0	Interlock1	BOOL	FALSE	TRUE (internal PLC bit): Interlock conditions for motion towards direction 1 are met. The object may move.
+8.1	Interlock2	BOOL	FALSE	TRUE (internal PLC bit): Interlock conditions for motion towards direction 2 are met. The object may move.
+8.2	Manual_Interlock1	BOOL	FALSE	TRUE (internal PLC bit): Interlock conditions for operator controlled motion towards direction 1 are met. The object may move.
+8.3	Manual_Interlock2	BOOL	FALSE	TRUE (internal PLC bit): Interlock conditions for operator controlled motion towards direction 2 are met. The object may move.
+8.4	Manual_Enable1	BOOL	FALSE	TRUE (bit, display device writing): set if the operator can control motions on-screen.
+8.5	Manual_Enable2	BOOL	FALSE	TRUE (bit, display device writing): set if the operator can control motions on-screen.
+8.6	Manual_Operation1	BOOL	FALSE	TRUE (bit, display device writing): Direction control key for direction 1 has been pressed at the display device.
+8.7	Manual_Operation2	BOOL	FALSE	TRUE (bit, display device writing): Direction control key for direction 2 has been pressed at the display device.
+10.0	Display_Order	ARRAY [0 to 15]	FALSE	TRUE (bit array, display device writing): Assignment of which motion is currently output at which screen position at the display device and of the corresponding hotkeys (top corresponds with Display_Order[0]. Only 1 bit can be TRUE at any one time.
*0.1		BOOL		
+12.0	Trigger1	BOOL	FALSE	TRUE (internal PLC bit): Triggers motion towards direction 1.
+12.1	Trigger2	BOOL	FALSE	TRUE (internal PLC bit): Triggers motion towards direction 2.

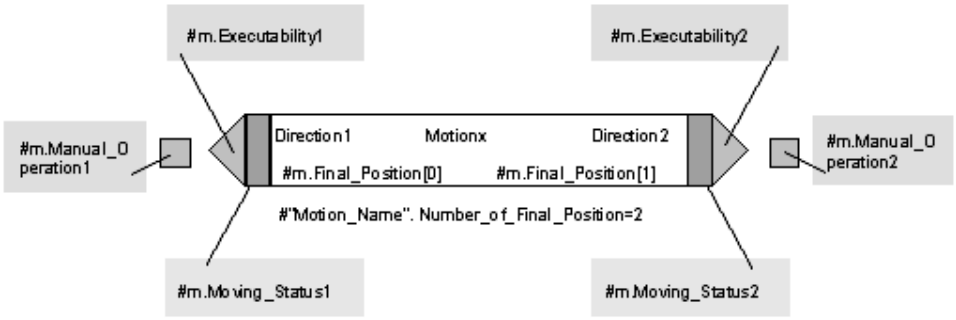
Address	Variable	Data Type	Initial Value	Comment
+12.2	Automatic_Trigger1	BOOL	FALSE	TRUE (internal PLC bit): Triggers motion towards direction 1 in auto mode.
+12.3	Automatic_Trigger2	BOOL	FALSE	TRUE (internal PLC bit): Triggers motion towards direction 2 in auto mode.
+12.4	Control1	BOOL	FALSE	TRUE (internal PLC bit): Sets the output for motion towards direction 1 in.
+12.5	Control2	BOOL	FALSE	TRUE (internal PLC bit): Sets the output for motion towards direction 2 in.
+14.0	Position_Flag	ARRAY [0 to 15]	FALSE	(internal bit array in the PLC): position flag for edge detection in reaction monitoring mode. Position_Flag[0] is assigned to Final_Position[0].
*0.1		BOOL		
+16.0	HMI_ID	DWORD	DW#16#0	Assigning the motion via the display device (as of S7-PDIAG V5.0 HF3)
=20.0		END_STRUCT		

6.7.11 Troubleshooting via UDT_Motion

Troubleshooting routines can be performed comfortably by means of hotkeys on the display unit, for example. Hotkeys are interconnected directly to the digital control inputs via the digital outputs of the display unit (e.g. wired or via DP interface) and thus enable instantaneous operator control of motions without any delay.

Requirement

Prerequisite is, however that you have used UDT_Motion for programming motions and have thus created a standardized interface for the motion screen. Motions are output on the operator panel in a semi-graphic format. The figure below shows a screenshot of UDT_Motion elements on the display unit.

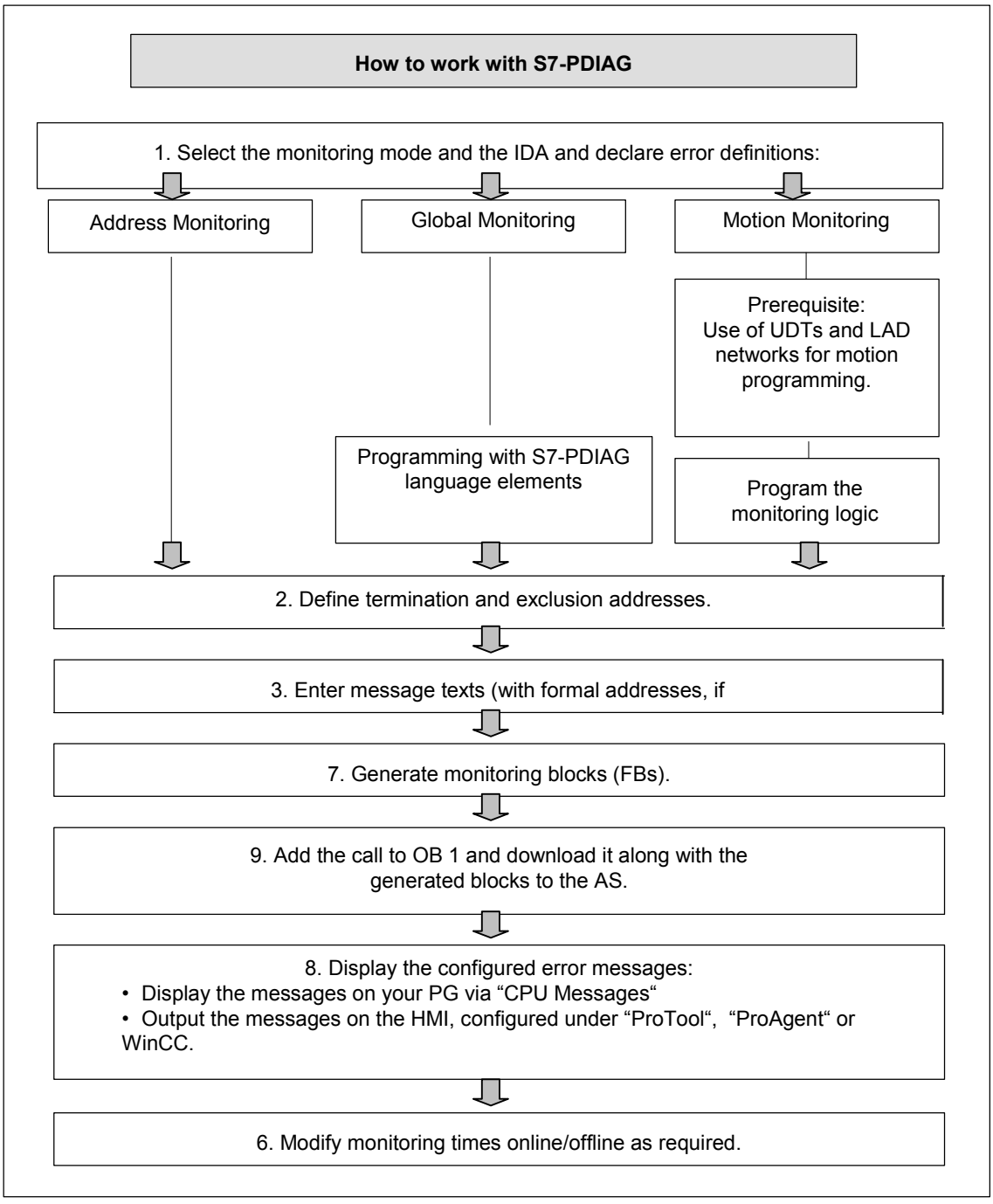


7 User Guide for S7-PDIAG

7.1 S7-PDIAG User Guide

This documentation shows you the basics of working with S7-PDIAG. The general procedure always remains the same, regardless of the error definitions you are going to create.

The following chapters provide a detailed, stepbystep description of the procedures for creating various monitoring definitions.


How to work with S7-PDIAG
1. Select the monitoring mode and the IDA and declare error definitions:
Address Monitoring
Global Monitoring
Motion Monitoring
Prerequisite:
Use of UDTs and LAD networks for motion programming.
Programming with S7-PDIAG language elements
Program the monitoring logic
2. Define termination and exclusion addresses.
3. Enter message texts (with formal addresses, if
7. Generate monitoring blocks (FBs).
9. Add the call to OB 1 and download it along with the generated blocks to the AS.
8. Display the configured error messages:
• Display the messages on your PG via “CPU Messages“
• Output the messages on the HMI, configured under “ProTool“, “ProAgent“ or WinCC.
6. Modify monitoring times online/offline as required.

Perform the following steps when working with S7-PDIAG:

1. In the first step, select a suitable monitoring mode and then create an error definition that precisely describes the process error status you want to monitor.
   - First, select the initial diagnostics address, irrespective of the set monitoring mode.
   - If you decide to use global monitoring, program the logic using the S7-PDIAG language elements.
   - if you decide to use motion monitoring, configure the monitoring logic in the displayed dialog.
2. Define termination and exclusion addresses as required.
3. In the next step, configure your error message texts.
4. After you have configured all error definitions and the corresponding message texts, you can generate the monitoring blocks that contain all data relevant for S7-PDIAG.
5. Now add a call for the error detection blocks at the end of the OB1 statement list or at the position required and then download this modified block as well as the monitoring blocks generated by S7-PDIAG to your automation system.
6. If an error occurs, the system outputs an error message to all connected display devices (e.g. PG or OP), showing the message text you configured.
7. You may also modify the set monitoring times via the "Modify Times" function, both in online or offline mode.

7.2 Selecting the Initial Diagnostic Address (IDA)

Before you create a monitoring definition, you first need to select an initial diagnostics address.

You can choose one of three methods to do so, depending on which tool you select the IDA from, e.g. the LAD/STL/FBD editor, the unit overview of S7-PDIAG or from the symbol table.

Selecting the IDA in the LAD/STL/FBD Editor:

1. Open the block in which you want to create a monitoring definition.
2. Click the address for which you want to define a monitoring definition. This address will be highlighted.
3. Open the "Process Monitoring Definitions" dialog box via the menu command **Edit > Special Object Properties > Monitoring**, or select the context-sensitive menu command **Special Object Properties > Monitoring** via right-click.
4. Select the monitoring definition you want to create in the "Templates" list box.
5. Click the "New" button. In the next dialog box you can create the monitoring definition according to your selection.
6. The selected address for which you want to create a monitoring definition is then entered in the "Initial Diagnostic Address" box of the "Definition" tab.
7. You can call further information on the input of data at this dialog box via the F1 key.

Selecting the IDA in the unit overview:

1. Select the object in the hierarchy for which you want to create a monitoring definition.
2. Select the menu command **Insert > Monitoring Definition**. In the next dialog "Process monitoring modes",
3. select the monitoring definition you want to create from the "Templates" list box.
4. Click the "New" button. In the next dialog box you can create the monitoring definition for the type you selected.
5. Enter the address for which you want to define a monitoring definition in the "Initial Diagnostic Address" field of the "Definition" tab.
6. You can call further information on the input of data at this dialog box via the F1 key.

Selecting the IDA from the symbol table:

1. Open the desired symbol table by double-clicking on it ("Symbols" object).
2. Select the symbolic name of the address for which a monitoring function is to be created.
3. Open the "Process Monitoring Definitions" dialog box using the menu command **Edit > Special Object Properties > Monitoring**, or call the context-sensitive menu command **Special Object Properties > Monitoring** via right-click.
4. Select the type of monitoring definition you want to create in the "Templates" list box.
5. Click the "New" button. In the next dialog box you can create the definition for the monitoring type you selected.
6. The selected address for which you want to create a monitoring definition is then entered in the "Initial Diagnostic Address" field of the "Definition" tab.
7. You can call further information on the input of data at this dialog box via the F1 key.
8. To exit the dialog box, click "OK" or "Close".

Notes

New monitoring definitions are found under the "Standard Group" directory in the unit overview of S7-PDIAG.
This function is only available as of STEP 7 V5.1+SP3.

7.3 Selecting the Monitoring Mode

There are two ways of selecting the monitoring mode for a new monitoring definition.

Selecting the monitoring mode in the LAD/STL/FBD Editor:

1. Select the initial diagnostic address in the block.
2. Open the "Process Monitoring Definitions" dialog box via the menu command **Edit > Special Object Properties > Monitoring**, or call the context-sensitive menu command **Special Object Properties > Monitoring** via right-click.
3. Select the type of monitoring definition you want to create in the "Templates" list box. The following options are available:
   - S7_PDIAG: Address Monitoring
   - S7-PDIAG: Global Monitoring
   - S7-PDIAG: Action Monitoring
   - S7-PDIAG: Startup Monitoring
   - S7-PDIAG: Reaction Monitoring
   - S7-PDIAG: Interlock Monitoring
4. Click the "New" button. In the next dialog box you can create the definition for the monitoring type you selected.
5. You can call further information on the input of data at this dialog box via the F1 key.

or

1. Open the "Diagnostics" tab and select any row, then select **New** in the context menu via a right-mouse click.
2. A new error definition with the last monitoring mode selected will be created. To change the monitoring mode, click on selection box in the "Mode" column of the "Diagnostics" tab and then select the desired monitoring mode.
3. Enter the desired initial diagnostic address.

or

1. Select the initial diagnostic address in the block.
2. Insert a new monitoring with the selected initial diagnostic address via the menu command **Edit > Special Object Properties > Create New Error Definition** or via the right-mouse button in the context menu **Special Object Properties > Create New Error Definition**.
3. A new error definition with the last monitoring mode selected will be created. To display it, open the "Diagnostics" tab.
4. To change the monitoring mode, click on the selection box in the "Mode" column of the "Diagnostics" tab and select the desired monitoring mode.

How to select the monitoring mode in the unit overview of S7-PDIAG:

1. Select the menu command **Insert > Monitoring Definition**. In the next dialog "Process monitoring modes", select the monitoring definition you want to create from the "Templates" list box.
   - S7_PDIAG: Address Monitoring
   - S7-PDIAG: Global Monitoring
   - S7-PDIAG: Action Monitoring
   - S7-PDIAG: Startup Monitoring
   - S7-PDIAG: Reaction Monitoring
   - S7-PDIAG: Interlock Monitoring
2. Click the "New" button. In the next dialog box you can create the definition for the monitoring type you selected.
3. You can call further information on the input of data at this dialog box via the F1 key.

How to select the monitoring mode in the symbol table:

1. Open the desired symbol table by double-clicking on it ("Symbols" object).
2. Select the symbolic name of the address for which a monitoring function is to be created.
3. Open the "Process Monitoring Definitions" dialog box using the menu command **Edit > Special Object Properties > Monitoring**, or call the context-sensitive menu command **Special Object Properties > Monitoring** via right-click.
4. Select the type of monitoring definition you want to create in the "Templates" list box. The following options are available:
   - S7_PDIAG: Address Monitoring
   - S7-PDIAG: Global Monitoring
   - S7-PDIAG: Action Monitoring
   - S7-PDIAG: Startup Monitoring
   - S7-PDIAG: Reaction Monitoring
   - S7-PDIAG: Interlock Monitoring
5. Click the "New" button. In the next dialog box you can create the definition for the monitoring type you selected.
6. You can call further information on the input of data at this dialog box via the F1 key.

7.4 Defining Address Monitoring

Procedure:

1. Open the "Address Monitoring" dialog box as follows:
   - Select the unit from the unit overview under which you want to insert the monitoring definition and then call the menu command **Insert > Monitoring Definition**. Select address monitoring in the next dialog box "Process Monitoring Definitions" and click on "New".

   or

   - select the initial diagnostics address at a block in the LAD/STL/FBD editor and right-click to open the "Process Monitoring Definitions" dialog box. Select "Address Monitoring" from the "Templates" list box and click on "New".

   or

   - in the symbol table, select the symbolic address for which you want to create a monitoring definition and right-click to open the "Process Monitoring Definitions" dialog box. Select "Address Monitoring" from the "Templates" list box and click on "New".
2. In the "Definition" tab, choose either to create a level monitoring or an edge monitoring definition. You can specify a delay time for the level or edge trigger and enable the monitoring definition as well as initial value acquisition.
3. Specify the general information on the monitoring definition in the "General" tab.
4. Define the message which is to be sent to the display device after an error has occurred.
5. Confirm your entries with "OK" or "Save".
6. You can call further information on the input of data at this dialog box via the F1 key.

Note

If you have initially created a monitoring definition for a FB for which an instance DB already existed, you must delete and recreate it in order to enable the diagnostic functions for the instance DB.

7.5 Defining Motion Monitoring

Procedure:

1. Open the "Process Monitoring Definitions" dialog box by selecting a motion in the unit overview and then selecting the menu command **Insert > Monitoring Definition**. You can set any of the following default motion monitoring definitions:
   - S7-PDIAG: Action Monitoring Definitions
   - S7-PDIAG: Startup Monitoring Definitions
   - S7-PDIAG: Reaction Monitoring Definitions
   - S7-PDIAG: Interlock Monitoring Definitions

   or:

   Select the initial diagnostic address in a block in the LAD/STL/FBD Editor and open the "Process Monitoring Definitions" dialog box via the menu command **Edit > Special Object Properties > Monitoring** or right-click. Select the required monitoring definition from the Templates" list box and click the "New" button.

or:

1. in the symbol table, select the symbolic address for which you want to create a monitoring definition and right-click to open the "Process Monitoring Definitions" dialog box.
2. Edit the default monitoring logic in the "Definition" tab according to your specific monitoring type. You can enable monitoring and initial value acquisition.
3. Specify the general information on the monitoring definition in the "General" tab.
4. Define the message which is to be sent to the display device once the error has occurred.
5. Confirm your entries with "OK" or "Save".
6. You can call further information on the input of data at this dialog box via the F1 key.

Notes

- When programming motions in your user program, we recommend you revert to the LAD Networks for Motion Programming and to UDT_Motion, both of which are provided by S7-PDIAG. If you do so, you can use the default motion monitoring logic and thus gain maximum benefits.
- If you have initially created a monitoring definition for a FB for which an instance DB already existed, you must delete and recreate it in order to enable the diagnostic functions for the instance DB.

7.6 Defining Global Monitoring

Procedure:

1. Open the "Global Monitoring" dialog box as follows:
   - Select a unit from the unit overview and call menu command **Insert > Monitoring**. In the next dialog box "Process Monitoring Definitions, select global monitoring from the "Templates" list box and then click on "New".

   or

   - In the LAD/STL/FBD editor, select the Initial Diagnostics Address in a block and use the menu command or right-click to open the "Process Monitoring Definitions" dialog box. Select "Global Monitoring" from the "Templates" list box and click on "New".

   or

   - in the symbol table, select the symbolic address for which you want to create a global monitoring definition and right-click to open the Process Monitoring Definitions" dialog box. Select the monitoring mode required from the upper list box and then click on "New".
2. Enter the monitoring logic in the "Definition" tab. Please adhere to the information in Language Elements of S7-PDIAG. You can also enable monitoring and Initial Value Acquisition.
3. Specify the general information on the monitoring definition in the "General" tab.
4. Define the message which is to be sent to the display device once the error has occurred.
5. Confirm your entries with "OK" or "Save".
6. You can call further information on the input of data at this dialog box via the F1 key.

Note

If you have initially created a monitoring definition for a FB for which an instance DB already existed, you must delete and recreate it in order to enable the diagnostic functions for the instance DB.

7.7 Programming the Monitoring Logic for Global Monitoring Definitions

You can programs your user-specific logic for global monitoring.

Note

Do not enter more than 64 addresses in your monitoring logic for a general monitoring definition.

Procedure:

1. Select the Initial Diagnostics Address and call the "Global Monitoring" dialog box, by:
   - calling menu command **Insert > Monitoring** in the unit overview a then selecting "Global Monitoring" in the next dialog. Select the corresponding IDA and then click on "New" if you want to create a new global monitoring definition, or select an existing global monitoring definition under "Existing Monitoring Definitions" and then click on "Edit".

   or:

   - Select the initial diagnostic address in a block in the LAD/STL/FBD Editor and open the "Process Monitoring Definitions" dialog box via the menu command **Edit > Special Object Properties > Monitoring** or right-click. From there, you can reach the "General Monitoring" dialog box by clicking on "New" or "Edit".
2. Enter your monitoring logic in the "Monitoring" input box of the "Definition" tab, or edit the displayed monitoring logic. To do this, use the language elements of S7-PDIAG.
3. Confirm your entries with "OK" after you have configured the monitoring logic.

7.8 Defining Termination Addresses

The use of auxiliary networks for criteria analysis will be aborted immediately when a termination address is reached.

You can select one of the following methods of defining termination addresses, depending on whether you are going to select this address in S7-PDIAG or in the LAD/STL/FBD editor.

How to define termination addresses in S7-PDIAG:

1. Select S7-PDIAG in the **Options > Termination Address** menu command.
2. Enter the address required under "Termination Address" in the next dialog box. This can either be a symbolic or absolute address.
3. Click on "Insert" to add the address to the existing termination address list.
4. If you want to delete the existing addresses, select the required in the list and then click on "Delete".
5. If you want to delete all addresses in the list, click on "Delete All".
6. The "Generated termination addresses" list box shows you a table of all generated termination addresses, as well as their corresponding type and generating system (e.g. S7-PDIAG or S7-HiGraph). You can delete S7-PDIAG termination addresses only via the corresponding FB interface.
7. Click on "Convert" in order to check whether a monitoring type (i.e. a function block) exists for the termination address in the corresponding DB. If such a monitoring type is found, the termination address in the DB will be assigned to the corresponding FB.
8. Click on "OK" to save your entries and exit the dialog box.

How to define termination addresses in the "LAD/STL/FBD" editor:

1. In SIMATIC Manager, double-click on the block at which you want to define as a termination address. The block will then be opened in the "LAD/STL/FBD" editor.
2. In the LAD/STL/FBD" editor, open the varible overview and the variable detail view for the desired parameter via a double-click.
3. In the variable detail view, check the desired area in the "Exclusion Addresses" column. By doing so, you have defined this address as the exclusion address for the process diagnostics.
4. Further information can be called by pressing the F1 key.

7.9 Defining Exclusion Addresses

You can define Exclusion Addresses.

The criteria analysis function hides those addresses and auxiliary networks you are working on if they have been assigned a "0" value (only in conjunction with ProAgent, Version 5.0 or higher).

You can choose one of two methods of defining exclusion addresses, depending on whether you are going to select this address in S7-PDIAG or in the LAD/STL/FBD editor.

How to define exclusion addresses in S7-PDIAG:

1. Select the menu command **Options > Exclusion Addresses** in S7-PDIAG.
2. Enter the required addresses under "Exclusion Address" in the next dialog box. This can either be a symbolic or absolute address.
3. Click the "Insert" button to add the address to the list of existing exclusion addresses.
4. If you want to delete existing exclusion addresses, select the required address in the list and click the "Delete" button.
5. If you want to delete all addresses in the list, click the "Delete All" button.
6. The "Generated Exclude Addresses" list box shows you a table of all generated termination addresses, as well as their corresponding type and generating system (e.g. S7-PDIAG or S7-HiGraph). You can delete S7-PDIAG exclude addresses only via the corresponding FB interface.
7. Click on "Convert" in order to check whether a monitoring type (i.e. a function block) exists for the exclusion address in the corresponding DB. If such a monitoring type is found, the termination address in the DB will be assigned to the corresponding FB.
8. Click on "OK" to save your entries and exit the dialog box.

How to define exclusion addresses in the "LAD/STL/FBD" editor:

1. In SIMATIC Manager, double-click on the block at which you want to define an exclusion address. The block will then be opened in the "LAD/STL/FBD" editor.
2. In the LAD/STL/FBD" editor, open the variable overview and the variable detail view for the desired parameter via a double-click
3. In the variable detail view, check the desired area in the "Exclusion Addresses" column. By doing so, you have defined this address as the exclusion address for the process diagnostics.
4. Further information can be called by pressing the F1 key.

7.10 Input of Message Texts

You can assign a message to a monitoring definition, which then appears on the display device when an error occurs. You can either enter the message text when you program the monitoring definition, or you can add or edit it at a later time.

The message text may contain formal addresses for use as associated values, which the system substitutes when it generates or displays this message.

Procedure:

1. Open the monitoring definition in which you want to enter or edit a message text.
2. You can enter the message text in the lower half of the dialog box.
   Options available:
   - Select the required priority class (1 to 16) for the message. The message priority indicates the importance of a message. Priority class 1 is the lowest. By assigning a priority, you can influence the format and order in which an error message is displayed on the operator panel.
   - Select the "Message Configuration" option if you want to define a user-specific message. In the "Text" input box, enter a message text with a max.string length of 254 characters.
   - Select the "Symbolic name" option if you want to set the name of the initial diagnostics address from the symbol table as message.

Note

The symbolic name will now be used as message text. However, if you modify the symbolic name at a later time, this will have no effect on the message text.

   - Select the "Symbol Comment" option if the symbol comment for the initial diagnostic address from the symbol table is to be used as the message.

Note

The symbol comment will now be used as message text. However, if you modify the symbol comment at a later time, this will have no effect on the message text. Formal addresses in the symbol comment will not be replaced.

   - Click the "Configure" button if you want more configuration options for the message. In this case you can define, for example different display devices, as well as the format of the displayed message.
3. Set the "With Acknowledgement" check box if errors are to be acknowledged by the operator. In this case the operator acknowledges the message by pressing a key on the operator panel.
4. Confirm your entries with "OK" or "Save" after you have defined your message as required.

7.11 Synchronizing Message Texts with Changed Symbol Table

You have configured a symbol name or a symbol comment as a message in S7-PDIAG.

If this symbol name or comment is subsequently changed in the symbol table, either manually or with the help of the foreign language tool, it is then necessary to synchronize the existing message texts with changed symbol table.

To do this proceed as follows:

1. Start S7-PDIAG from the SIMATIC Manager (menu command **Options > Configure Process Monitoring**) and open the project to which the changed symbol table belongs.
2. In S7-PDIAG, open the "Settings" dialog box (menu command **Options > Settings**). Go to the "Compile " tab and select the "Synchronize message texts to symbol table" check box.
3. Exist the dialog box with "OK".
4. To start the compilation so that the message texts are synchronized to the changed symbol table, select the **Process Diagnostics > Compile** or **Process Diagnostics > Compile All** menu command.

7.12 Formal Addresses in Message Texts

S7-PDIAG allows you to adapt message texts automatically to the corresponding instances during compilation and to embed associated values in the message text.

An associated value (or address) can be "appended" to a message text. This value is recorded by S7-PDIAG whenever the error is detected. The associated value is displayed on the operator panel (HMI) at the defined position in the message text. Enter an appropriate formal address in the message text.

Formal Addresses

The system provides several formal addresses which will be replaced either when it compiles the error definitions or displays the message.

7.13 Formal Addresses Substituted During Compilation

S7-PDIAG substitutes the following formal addresses during compilation:

$$u$$, $$u1$$	Name of the higher-level unit
$$u2$$ to $$u9$$	The name of the corresponding higher-ranking unit. **Example:** You create an error definition with the formal address $$u3$$ in the fifth subunit of the S7-PDIAG tree structure. The unit value is therefore "u" or "u1". Starting at this value, the counter increments until the higher-ranking unit u3 is reached. Unit u3 is then output to the HMI.
$$ur$$	Name of the top unit within the tree structure
$$m$$	Name of the motion Important note:
$$o$$	Diagnostic entry address of the instance error definition (symbolic format if no symbol exists in the absolute format)
$$d1$$	Name of the motion direction 1
$$d2$$	Name of the motion direction 2
$$a$$	Diagnostic entry address in absolute format
$$s$$	Diagnostic entry address in symbolic format
$$c$$	Symbol comment for the diagnostic entry addresses **Note:** In the message text you can replace the formal address with the symbol comment from the symbol table.
$$CpuName$$	Name of the configured CPU. If no CPU was configured, the formal address will be retained and is not substituted during compilation.

You can select the name source. Available are:

- The names of units
- The motion name
- The direction texts
- The name of the instance error definition of the initial diagnostic address

Please note:

- Formal addresses are replaced in the specific language used.
- The last component of the name is always used for units and motions. This means that the name is composed of the element located between the end and the first dot on the way to the beginning of the name.

Example:

Name	Result
Drilling_Machine.Feed	Feed
Drilling_Machine.Drilling_Machine.Feed	Drilling_Machine.Feed

Note

More than one formal address within the opening and closing characters ($$) is **not** permitted.

7.14 Formal Addresses Replaced in the Message Text

The system replaces the following addresses when displaying messages on the HMI:

You can define the position and format of an associated value in the message text by creating a description block for this value. Use this syntax:

- @1<element type><format definition>@

The associated value replaces this description block at this position in the message text.

Permissible Addresses for an Associated Value:

An associated value can be an address parameter of the type CHAR, WORD , INT, DWORD, DINT and REAL from the I, Q, M or DB area. You may also enter local tags as associated values in a function block that has an error definition. Depending on the type-instance concept, those values are expanded during compilation.

You may define the position and display format of the associated value for the message text.

Element type of associated values:

Sets a data type definition for an associated value.

Element type	Data type
X	BOOL, BYTE, WORD, DWORD, INT, DINT
C	CHAR
R	REAL

Format Types for Associated Values:

You can select one of the characters shown below to define the output format of an associated value. The format definition has a "%" prefix.

Format definition:	Description:	Display range:
%d	Decimal with preceding sign	-2147483648.. +2147483647
%u	Decimal without preceding sign	0..4294967295
%X	Hexadecimal	0..FFFFFFFF
%b	Binary	11111111111111111111111111111111
%[i]X	Hexadecimal number with i digits	
%[i]u	Unsigned decimal with i digits	
%[i]d	Signed decimal with i digits	
%[i]b	Binary number with i digits	
%[i][.y]f	REAL number	Signed value in the format [-]dddd.dddd dddd: one or several digits with "y" after the decimal point and a total number of digits "i".
%1s	Character	ANSI character

Examples of the description block for an associated value:

@1X%6d@: Display associated values in DEC format with max. 6 digits.

@1X%1b@: Display associated values in BOOL format with "0" or "1".

Example of a description block for CHAR or REAL data types:

@1C%1s@: Display associated value "A" in CHAR format.

@1R%4.2f@: Display associated value "5.4" as "5.40" in REAL format.

Note

To ensure the inclusion of associated values in your compilation, set the "Include associated values" check box on the "Compile" tab ("Customize" dialog box.). To open this dialog box, select the **Options > Customize > Compile** command.

7.15 Full Compilation of Error Definitions

In order to enable error monitoring in the active process, you must compile your error definitions for the active project in monitoring blocks.

You can either compile the error definitions completely or only the current changes.

The **Process Diagnostics > Compile Completely** menu command is **used to recompile all PDIAG blocks**.

Procedure:

1. Select the **Process Diagnostics > Compile Completely** menu command.
2. The "Compile" dialog box is then opened during the initial compilation run. You will be prompted to check the compilation settings before you start an initial compilation. Click on "OK".
3. In the "Defaults" tab of the next dialog box, enter the number of the monitoring blocks for error recognition and initial-value/status acquisition.
4. In order to call a user block you have programmed, enable the "Enable call" check box and enter its ID.
5. In order to use group priority, enable the "save" check box and enter the corresponding memory word.
6. Start compilation by clicking on "OK".

During the compilation run, the progress is shown in a progress bar. If the compilation run was unsuccessful, an error message is displayed.

Notes

- If the compilation run was unsuccessful, a list of the error definitions which have caused the errors is output.
- In order to avoid inconsistency, it is mandatory to update all ProTool/ProAgent configuration data.
- Please note that when you recompile all monitoring units, the time required may increase accordingly.

7.16 Compiling Changes in Error Definitions

In order to enable error monitoring in the active process, you must compile your error definitions for the active project into monitoring blocks.

You can either compile the error definitions completely or only the current changes.

The menu command **Process Diagnostics > Compile** initiates compilation only of **changes** made since the last generating run.

Procedure:

1. Select the menu command **Process Diagnostics > Compile**.
2. The "Compile" dialog box is then opened during the initial compilation run. You will be prompted to check the compilation settings before you start an initial compilation. Click on "OK".
3. In the "Defaults" tab of the next dialog box, enter the number of the monitoring blocks for error recognition and initial-value/status acquisition.
4. In order to call a user block you have programmed, enable the "Enable call" check box and enter its ID.
5. In order to use group priority, enable the "save" check box and enter the corresponding memory word.
6. Start compilation with "OK."

During the compilation run, the progress is shown in a progress bar. If the compilation run was unsuccessful, an error message is displayed.

Notes

- If the compilation run was unsuccessful, a list of the error definitions which have caused the errors is output.
- If you use the **Process Diagnostics > Compile** menu command to compile the current changes, warnings may be displayed which are not displayed during the next generation run. This always occurs if it was possible to create the monitoring blocks despite the warnings. In order to view all the warnings once again and thus to verify the complete program, you should perform a full compilation via the **Process Diagnostics > Compile Completely** menu command.
- After the compilation, a message displays the recompiled groups.

7.17 Update/Download of OB1

In order to process compiled monitoring blocks and to report defined errors, you must embed these in the sequence by inserting the call for the monitoring blocks at the end of OB1.

Procedure:

1. In SIMATIC Manager, open OB1 of the corresponding S7-PDIAG project with double-click.
2. Ad the following call for the error detection blocks:
   CALL FBxy, DBxy
   PDIAGZyklus:=#OB1_SCAN_1
   Note:
   Adapt the FB and DB numbers (here: xy) according to your block numbers.
3. Download OB1 to the PLC via the menu command **PLC > Download**.

Note

- It is best to add the call for the error detection blocks at the end of OB1, since all addresses have already been updated at this point in the cycle.
- If you have created multiple groups in your program, you only need to add the CALL for the standard group (FBxy, DBxy) error detection blocks to OB1. The CALLS for error detection of the groups you created are automatically added to the error detection FB of the standard groups.

7.18 Downloading Monitoring Blocks to the AS

In order to enable error monitoring during process runtime, you must download the generated monitoring blocks to the PLC.

Monitoring blocks represent the blocks for error recognition as well as for initial value and status acquisition, and are generated by S7-PDIAG based on the created error definitions . After you have downloaded these monitoring blocks to your user program, they perform the process diagnostic operations.

Procedure:

1. Select the menu command **PLC > Download** in S7-PDIAG or click the corresponding toolbar button.
2. You can also download the tasks to the AS via SIMATIC Manager.

Notes

- After you have initially downloaded the blocks to the CPU, you must also download the calling block (for example, OB1) in order to enable processing of the monitoring blocks. You can also download these data via SIMATIC Manager.
- All monitoring blocks will be embedded in one cycle, provided your CPU supports this action.

7.19 Display of Configured Messages

You can view configured messages via the STEP 7 function "CPU Messages".

Procedure:

1. Switch to online mode in the SIMATIC Manager.
   Result: The online project window pops up.
2. Select the corresponding S7-PDIAG program.
3. Call the function "CPU Messages" via menu command **PLC > CPU Messages ...**.
4. In the next "Settings" dialog window, enable the check box under "A", in order to be able to view Alarm_S messages. Close the dialog window.
5. After you have customized your error message display in "CPU Messages", you can trigger a process error in order to view the corresponding messages.

Before you can program process diagnostics for the operator panel (HMI), you need to implement the corresponding images in your S7-PDIAG project.

Further information is found in the documentation of your display device (HMI).

8 Welcome to the S7-PDIAG and ProAgent Sample Program for First-time Users

8.1 Getting Started with S7-PDIAG

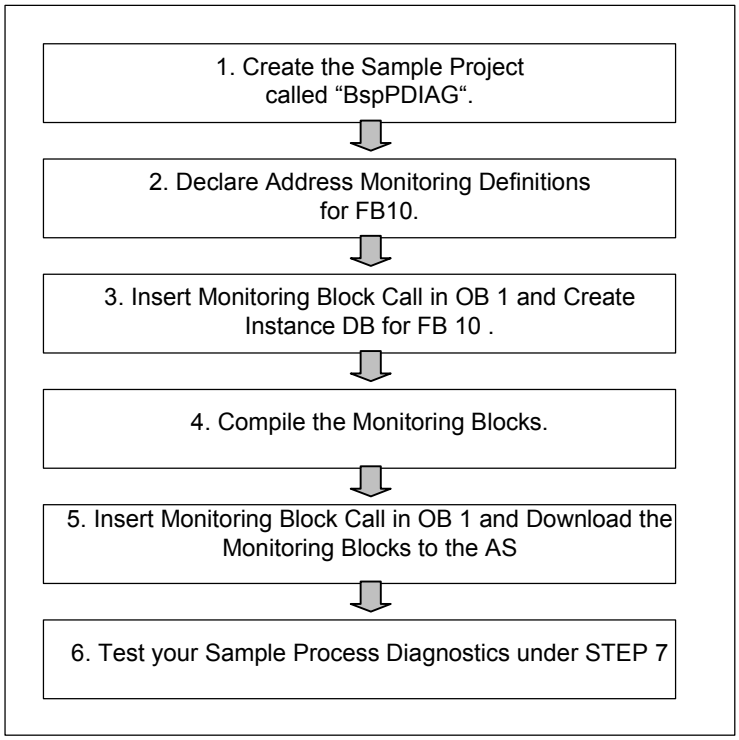
The example of address monitoring definition used in this chapter guides you step-by-step through an entire project configuration under S7-PDIAG.

We shall also show you all steps required under ProTool and ProAgent for programming a fully functional process diagnostics control and OP (HMI).

Overview of the Procedure

The diagram below provides and overview of procedures required for programming address monitoring definitions under S7-PDIAG.

Proceed as follows:



8.2 Create the Sample Project / Program

How to Create the Sample Project

In the first step, open the STEP 7 wizard in SIMATIC Manager and create a new project under the name "BspPDIAG". Add an S7 program to your corresponding HW Config.

How to Generate the S7 Sample Program

In SIMATIC Manager, select the block container for your "BspPDIAG" project below the S7 program and the hardware configuration. Call menu command **Insert > S7 Block > Function Block** and select the following function block:

- FB10

You can now use the blocks mentioned above to create an address monitoring definition.

Executability

In order to enable runtime for this sample program on the PLC, input byte "0" and output byte "1" must be interconnected to digital modules. If your system is only equipped with a CPU but not with digital modules, insert OB122 (I/Q access error) and monitor your parameters via "Status/Control variable".

Programming FB10

Open FB10 in SIMATIC Manager with double-click. Edit the statement list under "LAD/STL/FBD" as follows:

1. In the first network, enter:
   Network name: Logic operation Q1.0 in FB 10

   Program:
   ```
   A I 0.0
   A I 0.1
   A I 0.2
   A I 0.3
   = Q 1.0
   ```
2. Save the block via the file menu command **File > Save**.

8.3 Declaring Address Monitoring Definitions for FB10

Introduction

After you have programmed the block for your sample program, you can proceed to create an address monitoring definition for this block.

Procedure

1. If not already open, double-click on FB10 to open it in SIMATIC Manager. The "LAD/STL/FBD" Editor opens.
2. Output Q1.0 of the example is to be monitored. Address monitoring is therefore to be added for this output. Position the cursor on the instruction line "= Q 1.0" and call **Edit > Special Object Properties > Monitoring** to open the "Process Monitoring Definitions" dialog box.
3. In the "Templates" box, select "S7-PDIAG: Address Monitoring" and the click on "New".
   Result: The "S7-PDIAG: Address Monitoring" dialog box displays the "Definitions" tab. The initial diagnostics address displayed is taken from the statement list, i.e."Q1.0".
4. In order to assign this error message the corresponding message text, enter "Q 1.0 = Level 1 in FB10" in the "Message" group box .
5. Exit the tab with "OK". You have now configured an address monitoring definition for Q1.0 at level 1. This configuration is now displayed in the "Existing Monitoring Definitions" box of the "Process Monitoring Definitions" dialog box also.
6. Click on "Close" to exit the "Process Monitoring Definitions" dialog box.
7. Save the block via the menu command **File > Save,** for the newly created error definition to be saved in the block and then exit the LAD/STL/FBD Editor.
8. Insert the following call for FB 10 at the end of OB 1 in the "BspPDIAG" project:
   - CALL FB 10, DB 10
9. Click "Yes" In the subsequent dialog box to create the instance DB which does not yet exist (in this case: DB 10).
   Result: The DB 10 was created with the S7-PDIAG-relevant data and has also retained the attribute "S7_pdiag = true".
10. Save the block and its new error definition via menu command **File > Save** and exit the LAD/STL/FBD Editor.

8.4 Insert Monitoring Block Call in OB 1 and Download the Monitoring Blocks to the AS

Introduction

In order to enable the monitoring blocks you generated, you must download them to your PLC and add a call for these blocks in OB1, or at the appropriate point in your user program.

Requirement

You have generated the monitoring blocks for your entire user program.

Adding a Call to OB1

1. Open OB1 in the SIMATIC Manager by double-clicking it.
2. Add the following lines:
   CALL FB 44, DB 44
   PDIAGZyklus: = OB1_SCAN_1
3. Save the block and exit the "LAD/STL/FBD" Editor.

Note

FB 44 contains the error detection. If an error is detected in FB44 it will automatically call FB45, which is responsible for the acquisition of initial values and of the status.

Downloading the Sample Program

In SIMATIC Manager you can download the program example "BspPDIAG" to your PLC. Proceed as follows:

1. Select the block container in SIMATIC Manager.
2. Download the sample program to your CPU via menu command **PLC > Download > To Module.**

8.5 Generate the Monitoring Blocks

Introduction

The steps below show you how to generate monitoring blocks from your error definitions.

Procedure

1. In SIMATIC Manager, select the "Blocks" container and open S7-PDIAG via the menu command **Options > Configure Process Monitoring.**
   Result: The unit overview of S7-PDIAG displays the PDIAG-relevant units; in this case FB10 and DB10.
2. In S7-PDIAG, call menu command **Process Diagnostics > Compile**. If you are initially compiling these data, you will be prompted to check the compilation settings. Confirm this message with "OK".
3. In the "Defaults" tab of the next "Settings" dialog box that you can also call via menu command **Options > Settings**, set the error ID "44" for the error detection blocks to be compiled, and the ID "45" for initial value/status acquisition blocks.
4. Exit the dialog box with "OK". A progress bar is displayed and the monitoring blocks are generated. If an error occurs during compilation, a message will appear on the screen.
   Result: SIMATIC Manager displays the generated monitoring blocks and the corresponding SFCs required.

8.6 How To Test your Sample Process Diagnostics In STEP 7

Now that you have gone through the entire configuration process with S7-PDIAG using the example project, you can simulate a process error and display the configured messages via the STEP 7 function "CPU Message".

Requirements

In order to view the messages without using an OP, call the "CPU Message" function included in the standard software package. Proceed as follows:

1. Switch to online mode in SIMATIC Manager.
   Result: The online project window pops up.
2. Select the program example "BspPDIAG".
3. Call the "CPU Message" function via the menu command **PLC > CPU Messages ...**.
4. In the next dialog box "Customize", enable the check box below "A" in order to enable you to view the Alarm_S messages. Close the dialog box.

Now that you have customized your error message display under "CPU message", you can go ahead and trigger a process error.

How to Trigger the Error Message in FB10

Trigger a error message configured in FB10 as follows:

1. Set inputs I0.0, I0.1, I0.2 and I0.3, all at the same time. If not in possession of digital modules, you can use the STEP 7 function "Status/Control Variable".
   Result: You set output Q1.0 at FB10 to "1". S7-PDIAG recognizes this as error due to your error definition. An error message will appear with the message text you entered. The "CPU Message" window now displays this error message.

What Comes Next?

In the previous chapters you have learned step-by-step how to use S7-PDIAG to create a STEP 7 program with diagnostic functions.

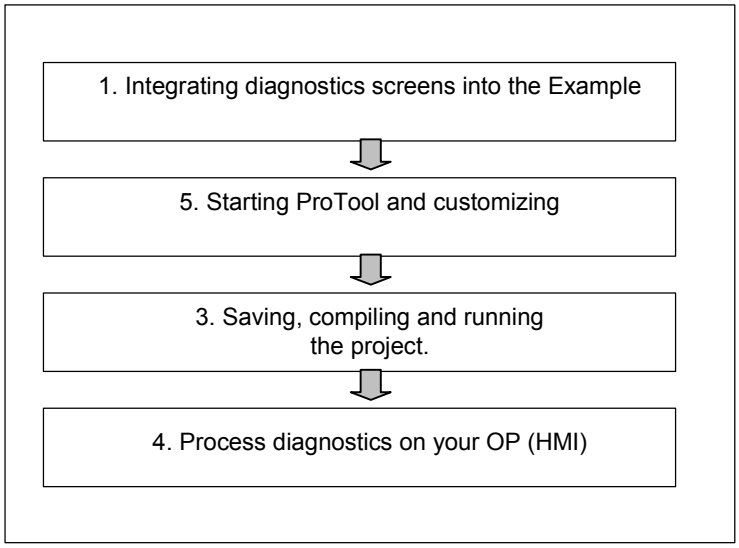
You will now learn how to create a configuration for diagnosing processes on an OP (hereafter referred to as operator panel) using the ProTool configuration software and the corresponding optional package ProAgent (from the SIMATIC HMI product family).

This section will then show you how to diagnose a process on the operator panel. This will also familiarize you with the different diagnostic screens.

8.7 Getting Started with ProAgent

In the next chapters will shall show you how to use ProTool to create a configuration with diagnostic functions for the sample project above, and how to download it to the OP and perform process diagnostics.

Proceed as follows:



Requirement

In order to configure process diagnostics under ProTool, you must have successfully generated the monitoring blocks for your user program, as described at the start of this chapter.

Operator Panel

The following description shows the OP25 as an example of a display device in all the figures. The procedure is identical for all display devices.

8.8 Integrating Diagnostics Screens into the Example

Introduction

Before you can program process diagnostics for the OP (HMI), you first need to implement the corresponding images into your S7-PDIAG project.

IN our example we only need the diagnostics screens, and you can therefore import the default configuration and diagnostics screen files supplied with the software package to your sample project. Otherwise you will have to copy and paste the diagnostic screens, as described in the manual.

Procedure

Integrate the diagnostic screens as follows:

1. If you have not already done so, start SIMATIC Manager and call menu command **File > Open**.
2. In the "Open" dialog box, select the "Projects" option. Select the "BspPDIAG" project from the list.
3. Similarly, open the "ProAgent" project.
   if this project does not appear in your table, click on "Find" and open the "ProAgent" project via the ProTool directory "Default\ProAgent".
   The "ProAgent" project contains default projects for various types of operator panels.
4. For our example we only need the diagnostics screens. You can therefore import the default configuration and diagnostics screen files directly to your sample project.
5. Drag and drop the "ProAgentPCmedium" configuration file to the "BspPDIAG" project or save via **File > Save as** to the "BspPDIAG" project.

8.9 Starting ProTool and Customizing

Introduction

The next step is to start ProTool and customize your settings. Of particular importance is that you select the network parameters, the CPU and the units.

How to Select Network Parameters and the CPU

Proceed as follows:

1. Start the ProTool CS configuration software by doubleclicking on the symbol for ProAgentPCmedium.
2. In the configuration overview, select the "PLC" item.
3. Double-click on "PLC_1" on the right side and then click on "Parameters" in the control dialog.
4. Select your network parameters and the networked CPU.
5. Confirm your entries with "OK".

How to Select Units

Now select these units for which you want to enable process diagnostics. Proceed as follows:

1. Call the menu command **PLC > ProAgent**.
2. Highlight "PLC_1" and click on "Select Unit". The entry will be included in the list of selected units.
3. Exit the "ProAgent" dialog box with "**OK**".
   Result: You have now enabled diagnostics functions for all units of PLC_1.

8.10 Saving, Compiling and Starting the User Program

Introduction

After you have completed the configuration, all that is left to do is to save, compile and start the user program. You can initiate these steps all at once by starting ProTool RT.

Note

If you do not want to use the same PC that you used for the configuration as the operator panel in this example, you must download the compiled program to the operator panel and then start it there.

Procedure

Proceed as follows:

1. Click on the symbol
2. Answer the compiler prompt with **Yes**.
3. **Result:** ProTool synchronizes its data to the STEP 7 database. This routine copies the diagnostic data and ALARM_S message texts from the database to the ProTool configuration.
   The save, compile and download sequence is now running. The status window meanwhile displays various messages on its "Compile" tab, e.g. ProTool messages during compilation.
4. ProTool RT starts up and you can now begin process diagnostics.

8.11 Process Diagnostics on your OP (HMI)

Introduction

Now that you created a configuration for the OP and downloaded it to your operator panel as described in the previous chapters, you can go on to perform a process diagnostics run.

Requirements

Before you can perform process diagnostics on the OP, you must have completed all the steps described in the previous sections:

- the control program must have been downloaded to the CPU and
- the configuration data must be in the operating unit.

Diagnostics Startup Screen

After you have started ProTool RT, the operator panel opens the diagnostics startup screen. One of your options here is to change either to the overview screen or to the message screen. Change for the time being to the message screen by clicking the corresponding button.

Message Screen

The message screen is blank at first because there have not been any errors so far.

1. Now simulate an error in FB10 as you have done previously.
   An error message is output on the message screen of the OP:
2. Click on *ACK* to hide the message window.
   Of course, although you have now acknowledged the message, you still have to react to the error itself. Until you clear the error, the small window containing the error character will keep flashing.

Explanations Relating to the Message Screen

The asterisk to the left of the message indicates that this message is diagnosable. Since this is the only message displayed so far, it is already selected. You recognize the selection by the inverted message line (light letters on a dark background).

When several messages are displayed, you first have to use the cursor keys or the mouse to select the message for which you want to perform process diagnostics. Then press the corresponding button to call the overview screen.

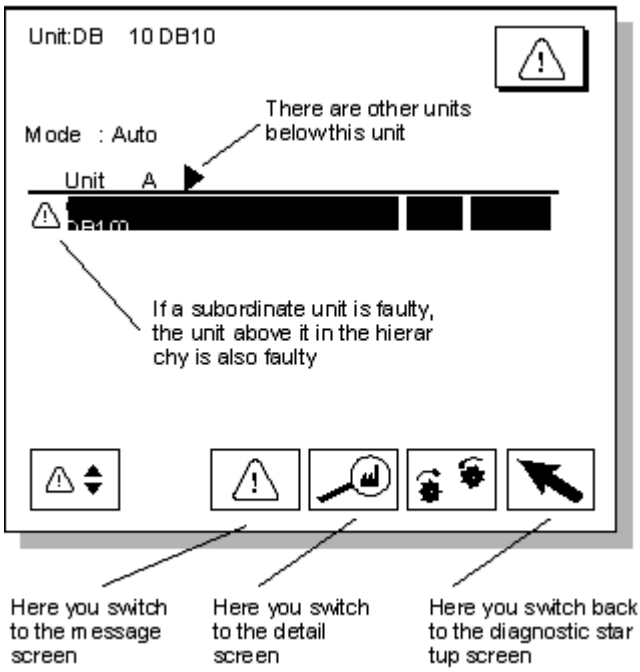
"Faulty Unit" Overview Screen

A warning triangle identifies the faulty unit. The warning triangle is flashing, since this is the first error to occur. If the error had involved consequential errors, warning symbols would likewise appear for the affected units, but they would not flash.

You recognize where the error first occurred from the location of the warning triangle. In many cases, this is where you will find the cause of the error and its consequential errors. The faulty unit has already been selected. Above the unit table you can see an arrowhead pointing to the left. This arrow indicates that the selected unit (Unit_11 in this case) is part of a unit at a higher hierarchy level.

"Higher-level Unit" Overview Screen

The following overview screen shows you the higher-level unit, in this case DB10.



Explanations relating to "Higher-level Units"

A warning triangle identifies the faulty unit. The warning triangle is flashing, as this is the first error to occur. If the error had involved consequential errors, warning symbols would likewise appear for the affected units, but they would not flash.

You recognize where the error first occurred from the location of the warning triangle. In many cases, this is where you will find the cause of the error and its consequential errors. The faulty unit has already been selected.

Now press the corresponding button to call the detail screen.

Detail Screen

The detail screen specifies the error triggering signals.

The signals that triggered the alarm message are identified by a lightning symbol. The address monitoring function we have defined in the *Getting Started with S7-PDIAG* chapter monitors output Q1.0. The error message will be triggered if Q1.0 = 1. This is the case in the current situation.

You can see the cause in the central area of the detail screen:

The status of all inputs I 0.0, I0.1, I0.2 and I0.3 is "1". According to the assignment, the status at output Q1.0 was therefore also set to "1". In order to eliminate this error, at least one of the inputs must be reset to "0".

Display as Signal List

In order to display the program code in the central area of the detail screen, you can toggle between STL, signal list and LAD by clicking on the corresponding button.

The display will appear as a symbol list in the central area of the detail screen.

LAD Display

Click again on the corresponding button. This will move you cyclically to the next display:

The display will appear as a ladder diagram in the central area of the detail screen.

Conclusion

You have now learned how to simulate an error, monitor its error message on the PC and locate the cause of error.

9 Ways of Defining Monitoring

9.1 Different Ways of Defining Monitoring Functions

Monitoring definitions can be configured at addresses in the symbol table, FCs and FBs.

You can define monitoring functions by:

- Configuring Monitoring Definitions in the Symbol Table
- Configuring Monitoring Definitions in FCs
- Configuring Monitoring Definitions in FBs

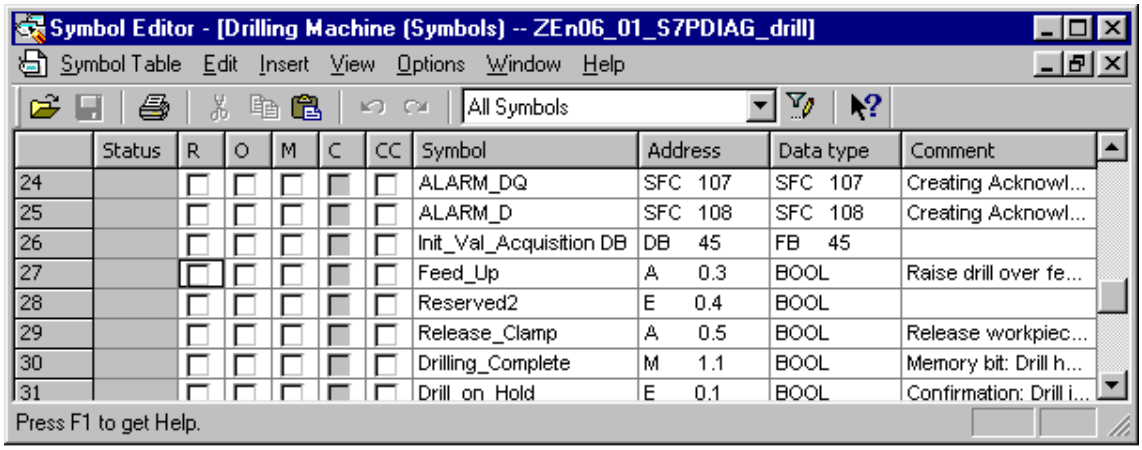
9.2 Configuring Monitoring Definitions in the Symbol Table

Requirements

You first need to enable the display of columns "R, O, M, C and CC" via the "View" menu command in order to configure monitoring definitions in the symbol table. The addresses to be monitored can then be defined in column "R".

Procedure

1. Set the check box in column "R" for the address to be monitored.



2. You can select the monitoring definitions in the selection dialog that appears next. Select the appropriate template, for example address monitoring and click on ”New”.
3. You can now configure the monitoring definition and enter the message text.

Note

- Disable initial value acquisition.
- The "Initial Diagnostic Address" box shows the symbol of the monitored address.
- Addresses no longer used in the user program can also be monitored.

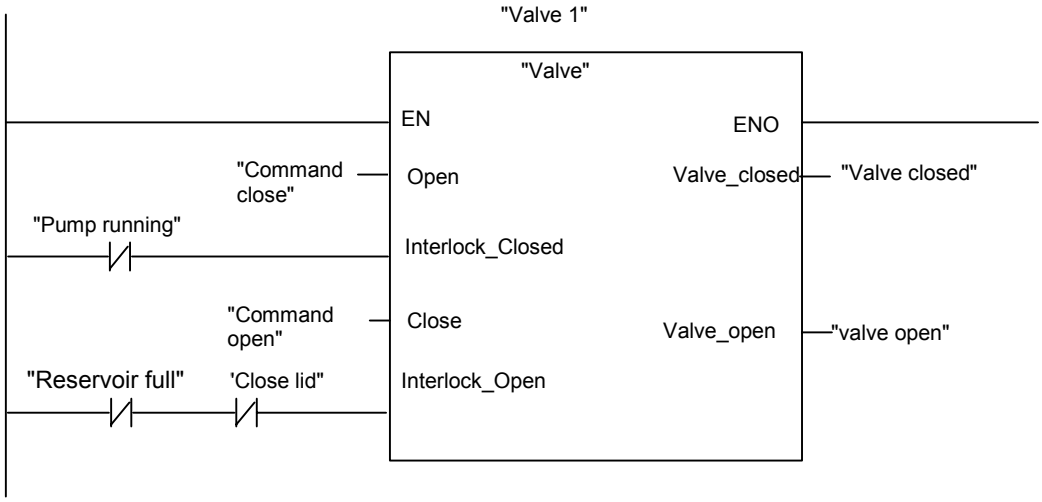
9.3 Configuring Monitoring Definitions in FBs

FB networks can be used also to monitor any addresses (conditions and/or assignments).

However, a significant difference has to be taken into account at the FBs. FBs are intended to be used at multiple instances (instantiated) and therefore formal parameters are used almost exclusively in the FBs as address. These formal parameters are assigned actual parameters when you instantiate the FBs in the LAD/STL/FBD Editor.

Under S7-PDIAG it is now possible to configure monitoring definitions for the formal parameters, thus allowing you to create standard blocks with integrated monitoring definitions (for example blocks for valves, motors etc.). If initial value acquisition is enabled for the monitored signals, the values of the corresponding actual parameters and the addresses will be stored in the upstream networks, rather than the values of the formal parameters.

Example:



When you monitor the "Interlock_Open" input and an error occurs, the status of the "Reservoir full" and "Close lid" addresses will be saved as a result of the enabled initial value acquisition function.

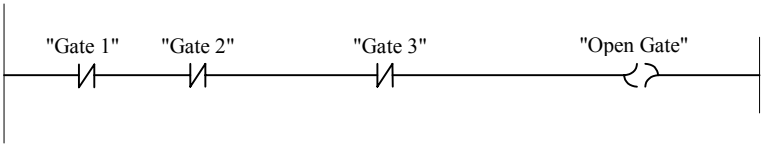
Note

S7-PDIAG offers formal parameters for displaying the name of the instance (e.g. valve 1) and the name of the station (= name of the higher-ranking block). This feature allows you to display both the error and its location in the error message.

9.4 Configuring Monitoring Definitions in FCs

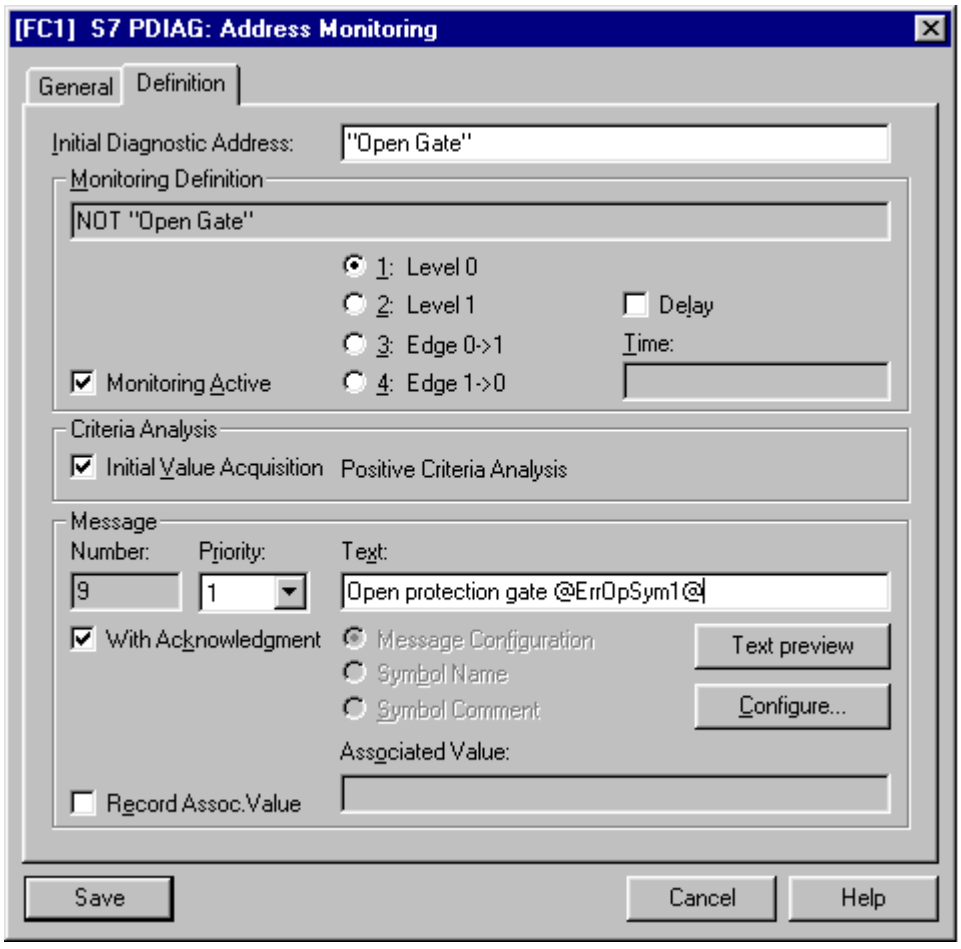
Any addresses (conditions and/or assignments) can be monitored in the network of FCs.

Example:



In this network, it is possible either to monitor each gate contact individually or the "Open Gate" group signal. S7-PDIAG can save the initial values after the monitoring function has been triggered, i.e. the status of the addresses used in the network, while ProAgent determines the triggering address by means of criteria analysis. This feature reduces programming work drastically down to configuring only a small number of monitored addresses.

This is what your configuration would look like, for example.



Level 0 is monitored in this case, since the level of the limit switch signal is ”0” when the protective guard is open (quiescent current principle). The triggering address is determined by means of initial value acquisition and a criteria analysis. The expression @ErrOpSym1@ determines that the triggering address appears in the error message line. For example, if gate 2 has been opened, this message text would be output: "Protective guard open Gate 2”.

Note

- You can configure monitoring definitions in blocks only if you set the "Address identifier" and "Process Diagnostics Address (S7-PDIAG)" check boxes in the "View" tab, under the **Options > Customize** menu command in the LAD/STL/FBD Editor. For monitoring blocks to be represented in color, the check boxes ”Address identifier” and ”Process Diagnostic Address” (S7-PDIAG) in the "View" tab under the **Options > Customize** menu command in the LAD/STL/FBD Editor have to be checked.
- Before you configure a monitoring definition for an address, you first need to select this address in the LAD/STL/FBD Editor. There you can right-click on the "Special object properties” line to open the context sensitive "Monitoring" menu command and then call the selection dialog of S7-PDIAG.
- When monitoring an address that belongs to an assignment (coil in the function chart, initial value acquisition must be enabled in order to allow ProAgent to locate the triggering signal in the network by means of criteria analysis).

10 Defining Monitoring Templates

10.1 Creating User Templates

In addition to the default monitoring definitions supplied with S7-PDIAG, you can also create user templates for specific monitoring definitions. The use of such templates means that configuring is much easier and requires less time and effort. You can also store incomplete monitoring logic in your own templates.

Note

Note that you are only storing a message template, and not the entire message within a template. This is why no message number is assigned in a template. However, you can configure the message text for your monitoring definition in the template.

Proceed as follows:

1. Select the menu command **Options > Templates** in S7-PDIAG.
2. In the "Templates" or "Existing Monitoring Definitions" tab of the next dialog box, select the appropriate monitoring definition and click on "New".
   Note: Existing monitoring definitions are only displayed if you have set an initial diagnostics address.
3. The "Template (...)" dialog box opens. Make your entries in the "General" and "Definition" tabs.
4. Click on "Save" to save your template.
5. Your template will be inserted in the "Template" dialog box, including the name of the monitoring type. Error definitions previously generated on the basis of the template will not be affected if you modify the template at a later time.
6. You can use this template for all new monitoring definitions; select it and click on "New".

11 Creating and Editing Objects

11.1 Creating Projects

You cannot create new projects under S7-PDIAG. However, you can create new projects in SIMATIC Manager:

- Using the Wizard
- Manually.

How to Create a Project Using the Wizard

Proceed as follows:

1. Call **File > Wizard 'New Project'** in SIMATIC Manager.
2. In the dialog boxes, enter the information requested by the Wizard.

How to Create a Project Manually

Proceed as follows:

1. In SIMATIC Manager, select **File > New**.
2. Select the "User projects" tab in the "New Project" dialog box.
3. Verify that the entry "Projects" is selected from the drop-down list "Type:".
4. Click on the "Insert in current multiproject" check box if you want to insert the project at this location. The check box is enabled only after you have selected the multiproject from the project window.
5. In the "Name:" input box, enter the name of the new user program. You can also select a name from the list above in order to use it as basis for a new project name.
6. Click on "OK" to create the new project.

11.2 Opening Projects

How to open an existing project in S7-PDIAG:

1. Call menu command **Process Diagnostics > Open** or click the corresponding toolbar button to open the "Open" dialog box.
2. Select the project in which you want to configure your monitoring definitions. Navigate to the required S7 program and click on the "Blocks" directory.
3. Confirm with "OK."

11.3 Address Priority in Projects

In the STEP 7 SIMATIC Manager, you can open a dialog box in which you can determine the properties of the block folder, i.e. whether to use the symbol or the absolute value to open blocks after changes at the symbol table.

If you have set symbol priority at the block folder in SIMATIC Manager, i.e. you have assigned them an address priority and you have deleted or modified a symbol in the symbol table and thus its assignment to an absolute value, the "Address priority" dialog box opens automatically when you call S7-PDIAG.

The "Address priority" dialog box in this case lets you once again assign a symbol to existing absolute values.

11.4 Copying Projects and Blocks

How to Copy a Project:

1. In SIMATIC Manager, select the project to be copied and
2. Call menu command **File > Save As**.
3. In the "Save As" dialog box, decide whether you want to save the file with or without reorganization. When working on older or frequently modified projects you should always select "Save with Reorganization", since in this case the system will optimize the database and check the project structure.
4. In the "Save Project As" dialog box, specify the name of the new project and, if required a storage location. Confirm with "OK."

When using SIMATIC Manager to copy projects or blocks containing S7-PDIAG data, note the following points:

- When you have created groups and copy the corresponding blocks, the group information is **not** included. The group information is **not** included either when you select to copy all the blocks in the "Blocks" directory.
- The group information is only included if you select and copy the complete "Blocks" directory or the higherlevel directories.
- Global error definitions which were created directly at the "Standard Group" into S7-PDIAG are only copied if the "Blocks" directory or the higherlevel directories are copied into the SIMATIC Manager.

11.5 Copying Error Definitions

You can copy error definitions and then paste them into the same unit and continue editing.

Proceed as follows:

1. Select the error definition you want to copy from the unit overview.
2. Copy the error definition as follows:
   - Select the menu command **Edit > Copy**, or
   - right-click and select "Copy" from the pop-up menu.

Note

After copying, the error definition is located in the clipboard and can be pasted via the menu command **Edit > Paste**.

11.6 Pasting Error Definitions

You can insert error definitions from the clipboard.

Proceed as follows:

1. Select the required object in the hierarchy into which you want to paste the error definition from the clipboard.
2. Paste the error definition as follows:
   - Select the menu command **Edit > Paste**, or
   - Right-click and select "Paste" in the pop-up menu.

Note
You can copy and paste error definitions only within the same hierarchical object.

11.7 Deleting Error Definitions

You can delete error or monitoring definitions you no longer require. There are three ways of doing this:

Deleting error definitions via the unit overview:

1. In the unit overview, select the error definition you want to delete.
2. How to delete the monitoring definition:
   - Select the menu command **Edit > Delete**, or
   - right-click to call the context-sensitive menu command "Delete".

How to delete error definitions via the LAD/STL/FBD Editor:

1. Select the initial diagnostic address of the monitoring definition in the block.
2. Open the "Process Monitoring Definitions" dialog box as follows:
   - Select the menu command **Edit > Special Object Properties > Monitoring**, or
   - right-click to call the context-sensitive menu command "Special Object Properties".
3. Select the error definition you want to delete from the lower list box.
4. Click the "Delete" button.

or

1. Open the "Diagnostics" tab in the LAD/STL/FBD for the corresponding block.
2. Select the error definition you want to delete by clicking the first column. The entire row will be colored as a result.
3. Call the menu command **Delete** in the context menu via a right-mouse click.

How to delete error definitions via the symbol table:

1. In the symbol table, select the monitoring definition address you want to delete.
2. Open the "Process Monitoring Definitions" dialog box as follows:
   - Select the menu command **Edit > Special Object Properties > Monitoring**, or
   - right-click to call the context-sensitive menu command "Special Object Properties".
3. Select the error definition you want to delete from the lower list box.
4. Click the "Delete" button.

Note

- When you delete a monitoring definition in a function block, the monitoring definition in the corresponding instance data block is not deleted. In the same way, if you delete the monitoring definition in the instance data block, it is not deleted in the function block. You should therefore always delete both monitoring definitions (in the function block and in the instance data block), so that no new monitoring definition is generated the next time the blocks are compiled, and to avoid monitoring definitions remaining in instance data blocks when no monitoring definitions exist in the corresponding function blocks.
- Error definitions can also be deleted if S7-PDIAG is not installed.

11.8 Editing Monitoring Definitions

Monitoring definitions which you have already created can still be modified at a later time.

Proceed as follows:

1. Open the monitoring definition you want to change as follows:
   - Select the monitoring definition in the unit overview. Open the corresponding dialog box via right-click or call the menu command **Edit > Object Properties**

   or:

   - Select the initial diagnostic address at a block via the LAD/STL/FBD Editor and open the "Process Monitoring Definitions" dialog box via menu command **Edit > Special Object Properties > Monitoring** or right-click. Highlight the monitoring definition to be modified and click on "Edit".

   or

   - Open the ”Diagnostics” tab in the LAD/STL/FBD Editor in a block and select the corresponding monitoring in the first column. Then open the context menu via a right-mouse click and select the menu command Object Properties or edit the boxes in the table.

   or:

   - in the symbol table, select the symbolic address for which you want to create a monitoring definition and right-click to open the Process Monitoring Definitions" dialog box. Select the monitoring definition you want to modify and click the "Edit" button.
2. You can edit the general information on the monitoring definition in the "General" tab.
3. You can edit the exact definition in the "Definition" tab and the type of monitoring definition (depending on the monitoring logic), and you can modify the delay time and enable or disable monitoring and initial value acquisition . You can also edit the message which is to appear on the operator panel after an error has occurred.
4. Click on "OK" or "Save" to confirm your entries and save the changes.
5. You can call further information on the input of data at this dialog box via the F1 key.

Note

In order to apply your changes to monitoring definitions to the process monitoring function as well, you must generate the monitoring blocks once again. Exception: If you have modified only the message text for the specified error event you do not need to recompile the monitoring blocks.

11.9 Grouping Units

S7-PDIAG lets you to group units via the menu command **Options > Group Units**.

Alongside with the standard group created by default, you can form up to 15 groups containing any number of units.

If a group created by you is deleted, the function automatically returns its units to the default group.

Monitoring blocks are created for each group during compilation. Of advantage are here the smaller blocks and shorter generating times.

Note

You should not group the units until you have completed your program hierarchy in the structuring phase.

Procedure:

1. In S7-PDIAG, select the **Options >** Group Units menu command.
2. The next dialog box displays the default group (0) under "Group". The left-hand list box displays the units belonging to the standard group.
3. Click on ”New" to create a new group. The dialog box "Group definitions" opens.
4. Enter a name for the new group to be created as well as the ID for the monitoring blocks for error recognition and initialvalue acquisition. Exit the dialog box by clicking on ”OK".
   Result: You have created a group that is displayed under "Group" in the "Group Units" dialog box.
5. In the "Group Units" dialog box, select the units required from the default group and move them to the new group by clicking the corresponding buttons (→, All >). You can also use the "*" and "All" buttons to return units to the default group.
6. In order to change the group defaults for the selected group again click on the ”Edit" command button.
7. You can use the ”Delete" command button to delete the selected group, with the exception of the permanently available standard group. The units assigned to this group will be returned to the standard group.
8. After you have made all settings, exit the dialog box with ”OK".
   Result: In the unit overview of S7-PDIAG, the new group is identified by a small numeral on a red background.

Note

- When you have created groups and then copy the corresponding blocks, the copy does **not** include the group information. The group information is **neither** included if you select and copy all blocks from the ”Blocks" directory.
- Group information is only included if you copy the complete "Blocks" directory, since this contains the relevant data.

11.10 Moving Units

S7-PDIAG offers you in its unit overview the option of moving units according to your system hierarchy.

These moved units will be identified by an icon and displayed in the virtual unit overview.

If the virtual hierarchy is unselected, instances can only be moved to other groups by means of the "Grouping" dialog.

You can call menu command **View > Unit Overview** to enable or disable the virtual unit overview.

12 Finding and Editing Objects

12.1 Find and Edit Objects

You can find and then edit error definitions, other error definitions, units, motions and templates of S7-PDIAG.

How to Find Objects

Proceed as follows to search for the above objects:

1. In S7-PDIAG select the **Options > Find** menu command.
2. In the subsequent "**Find In - ...**" dialog box specify what you want to find. If you want to find units or error definitions, specify whether you want to find:
   - Types (FB)
   - Instances (DB) or
   - Others (OB, FC, Global).

   Sensible defaults are always offered for the search.
3. If subordinate objects are to be included in the search, set the corresponding check box.
4. Click on "**Start**" to initiate the search, starting at the selected object in the unit overview.

Result: All objects found will be shown in a list.
The list starts on the left with the name of objects found, followed by information relevant to the object type, e.g.: initial diagnostics address, monitoring definition, initial value acquisition, acknowledgement, priority, display class and the message type of the object found.

The search result also displays the respective message ID for objects found at **instance error definitions** and **other error definitions**, thus making it much easier to assign messages at the display device.

Included is the number of found and selected objects.

How to Edit Objects

How to edit objects found:

1. Select the objects you want to edit. You have various options:
   - Select **one or more objects.** Click on "**Edit**" to open a dialog box according to your selected object, e.g. a unit, a motion or an S7-PDIAG error definition and edit the previous settings.
   - You can also right-click to open the context-sensitive menu. Call the "**Edit**" dialog box and modify your settings.
   - If you have selected only **one object**, you can view and edit it in the object properties dialog box by calling the dialog via the context-sensitive menu or by double-clicking the object.
   - If you have selected an **instance error definition**, you can view and edit the corresponding monitoring type via the context-sensitive menu.
   - The search result also displays the respective message ID for objects found at **instance error definitions** and **other error definitions**, thus making it much easier to assign messages at the display device.
2. Exit the respective "**Edit**" dialog box by using "**OK**" in order to save your settings.

13 Displaying and Editing Process Diagnostic-Specific Data

13.1 Displaying and Editing Monitoring Types

The unit overview of S7-PDIAG and the "Search in - ..." dialog box can be used to view and edit instance error definitions for the corresponding Monitoring Modes.

Proceed as follows:

1. In the unit overview select the instance error definition for which you want to have the corresponding monitoring type displayed or which you want to edit.

   or

   in the search result list of the "Find in - ..." dialog box, select the instance error definition whose corresponding monitoring type you want to view or edit.
2. Use the **Edit > Monitoring Type** menu command or the context-sensitive menu (right-click) to have the monitoring type belonging to the selected instance error definition displayed and edit it.
3. Save your changes by clicking on **OK**.
   Result: Changes become effective after the next compilation. All accessible instance error definitions belonging to the monitoring type will be overwritten accordingly.

Note

Already edited or modified elements of instance error definitions will not be overwritten when you recompile the data. If you want to overwrite all instance error definitions, set the appropriate option in the "**Options > Customize > Compile**" dialog box.

13.2 Display of Reference Data Generated by S7-PDIAG

After you have successfully generated reference data, S7-PDIAG saves them to the STEP 7 database. STEP 7 provides appropriate functions for displaying and filtering reference data.

Generated Reference Data:

The reference data created by S7-PDIAG encompass the data listed in the table below:

Reference data:	Displayed in:
Calls of the error recognition blocks with data block	Cross-reference list and program structure
Calls of the initial-value acquisition blocks with data block	Cross-reference list and program structure
Call of the user block with data block	Cross-reference list and program structure
Assignment of the memory word (storage of the group error priority)	Cross-reference list and assignment plan

Prerequisite for Generating Reference Data

The reference data for initial-value acquisition and error recognition blocks are created automatically during compilation. Reference data for the call of user blocks or for storing group error priority data will only be generated if you have enabled this function in the "Default” tab of the "Settings” dialog box.

Displaying and Filtering Reference Data

Procedure:

1. In S7-PDIAG, select the **Options >** Reference Data menu command.
2. The commands in the next menu can be used to:
   - Display the reference data or
   - Specify filtering conditions for displaying reference data.
3. For detailed information on further procedures refer to your current STEP 7 programming manual.

13.3 Update of Diagnostic-relevant Network Data

Previously, the system compiled diagnostic-relevant network data and wrote them to the S7-PDIAG database. These data were then modified and transferred to the OPs, depending on the respective exclusion addresses.

The result was that you also had to update display device data each time you performed an S7-PDIAG compilation.

S7-PDIAG V5.0 and higher now allows you to write diagnostic-relevant network data to the initial-value acquisition blocks generated by S7-PDIAG (under consideration of the exclusion addresses). The OPs (HMI) can thus request data from this location as required.

Of advantage in S7-PDIAG is that you only have to compile the changes and do not need to update the OPs (HMI) in each case.

Notes

- When you add or delete error definitions or edit existing error definitions, a message is displayed after the generation run informing you whether and if so which data have to be updated for the display devices (HMI).
- This form of saving diagnostic-relevant network data is particularly suitable for the commissioning phases. After you have completed commissioning, we recommend you save diagnostic-relevant network data to the display device (HMI) with respect to performance and the memory resources.
- The functions described above are not supported by ProTool/ProAgent earlier than V5.2+SP2.

Notes on modification of LAD/STL/FBD blocks:

- If you change LAD/STL/FBD blocks in the editor, **no** information is displayed after compilation indicating whether diagnostic-relevant network data for the display devices (HMI) are to be updated or not. These data do not have to be updated after changes within networks.
- When you change the name or number of networks, an incorrect name or number will be displayed at the display device for this network. However, the criteria analysis is correct. In this case it is up to you whether you update the diagnostic-relevant network data for the display devices.
- If you delete networks which contain multiple assignments or add networks which create multiple assignments (which thus contain an ambiguous address), you always have to update diagnostic-relevant network data for the display devices (HMI), otherwise you cannot perform a criteria analysis.

13.4 Generating Instances for Error Definitions

When generating instances, the system generates instance-specific monitoring definitions for the instance data blocks using a monitoring definition template you have created for a function block. You cannot edit such monitoring definitions until the instances have been generated.

Procedure:

1. Select the menu command **Process Diagnostics > Generate Instances**. The "Generate Instances" dialog box opens.
2. A progress bar in this window shows the progress when you generate the instances.

Note

If the generation of instance-specific monitoring definitions was unsuccessful, a message list opens to display the errors.

14 Configuring Process Diagnostic-Specific Message Texts

14.1 Presetting Message Texts

You can preset various options in the source for the message text that is to appear on the display device after an error has occurred. The default setting is enabled the next time you create a new error definition.

Proceed as follows:

1. Call the menu command **Options > Customize** to open the "Customize" dialog box.
2. Select the option you want to preset as a message text in the "General" tab.
3. Confirm with "OK."

15 Modifying Times

15.1 Modify Times Online

You can "Change Times" (not equal to 0) in monitoring definitions without having to recompile the monitoring blocks. Since you can do this both in online and offline mode, you are given the advantage of determining the monitoring time step- by-step.

Using the function "Change Times Online", you can change the monitoring times online in monitoring definitions without having to generate the monitoring blocks again each time, since the function automatically applies the changes made to the S7-PDIAG database and to the offline blocks.

Requirements:

The monitoring definition to be modified:

- Has already been implemented in the S7-PDIAG monitoring blocks during a previous compilation, and the blocks have been downloaded to the CPU
- Is active
- Contains a monitoring time which is unequal to "0."

Procedure:

1. Select the compiled monitoring definition and click on menu command **Edit > Modify Times Online** or the corresponding toolbar button.

   or:

   - Modify the times online from the LAD/STL/FBD Editor, if you are working online. Open the block linked to the monitoring definition and position the cursor on the initial diagnostic address. Select the context-sensitive menu command **Special Object Properties > Monitoring**. Select a monitoring function in the next dialog box and click on "Modify Time".

2. The selected monitoring definition is displayed in the next dialog box. However, you can only change the monitoring time.
3. Enter the new monitoring time and click on "Download". The modified monitoring times will be downloaded to the online blocks in the CPU.

Note

Your online data in the CPU are now inconsistent with offline data. In order to re-establish data consistency after you have modified the monitoring time online, update your offline data.

---

4. Click on "OK" to update the offline blocks in the S7-PDIAG database.

If the menu command or the toolbar button are disabled, this may be due to the following:

- You have selected a monitoring definition with an inconsistent online and offline database. In this case, once again download the monitoring blocks to the CPU and retry.
- You attempted to select an error definition in a function block.
- You attempted to change a monitoring function for which no monitoring time is defined.
- You are working on a project under S7-PDIAG Version 3.0. You must recompile this project in order to convert the monitoring blocks to V5.0, which you can then edit via Modify Times.

15.2 Modifying Times Offline

You can "Change Times" (not equal to 0) in monitoring definitions without having to recompile the monitoring blocks. Since you can perform this operation both in online and offline mode, you have the advantage of being able to determine the monitoring time step- by-step.

You can use the "Modify Times Offline" function to modify monitoring time definitions without the need to recompile the monitoring blocks after each change, since the function will automatically enter these modifications both in the S7-PDIAG database and in the offline blocks.

Requirements:

The monitoring definition to be modified:

- was implemented by a previous compilation in the S7-PDIAG monitoring blocks
- is active
- contains a monitoring time unequal to "0."

Procedure:

1. Select the compiled monitoring definition and click the menu command **Edit > Modify Times Offline** or use the corresponding toolbar button.

   or:

   - Modify times offline from the LAD/STL/FBD Editor, if you are working offline. Open the block linked to the monitoring definition and position the cursor on the initial diagnostic address.
     Select the menu command **Special Object Properties > Monitoring** via the context-sensitive menu. Select a monitoring function in the next dialog box and then click on "Modify Time".
2. The next dialog box displays the monitoring definition. However, you can only change the monitoring time.
3. Enter the new monitoring time and click on "OK" to write your changes to the offline database of S7-PDIAG.

Note

Your online data in the CPU are now inconsistent with offline data. If you want to make re-establish data consistency after you have modified the monitoring time offline, you once again need to download the modified monitoring definitions to the CPU.

---

4. Download the modified monitoring definitions to the CPU via the menu command **PLC > Download**.

If the menu command or the toolbar button are disabled, this may be due to the following:

- You have selected a monitoring definition with inconsistent online and offline database. In this case, download the monitoring blocks to the CPU and retry.
- You attempted to select an error definition at a function block.
- You have attempted to change a monitoring function for which no monitoring time is defined.
- You are working on a project under S7-PDIAG Version 3.0. You must recompile this project in order to convert the monitoring blocks to V5.0, which you can then edit via Modify Times.

16 Editing Attributes

16.1 Editing Unit Attributes

You can display and edit the attributes of units displayed in the unit overview.

Proceed as follows:

1. Select the unit or motion whose attributes you want to display or edit on the left-hand side of the unit overview.
2. Open the "Properties" dialog box as follows:
   - Select the menu command **Edit > Object Properties**, or
   - Right-click and select the "Object Properties" command.
3. In the "Properties" dialog box, you can view the properties of the unit in the "General" and "Attributes" tabs.
4. Modify the attribute values as required.
5. Confirm your changes with "OK."

Note

You can find a description of the various attributes in the online help by pressing F1 or by clicking the "Help" button in the "Attributes" tab of the "Properties" dialog box.

16.2 Editing Motion Attributes

You can display and modify the attributes of motions displayed in the unit overview.

Two options are available: You can either display the motion attributes in graphical format on the motion screen or in text-based format in the "Properties" dialog box.

Proceed as follows:

1. Select the motion whose attributes you want to display or edit on the left-hand side of the unit overview.
2. Open the "Motion Screen" dialog box as follows:
   - Call menu command **Edit > Motion Screen**, or
   - right-click and select the command "Motion Screen".
3. Modify the attribute values you require.
4. Confirm your changes with "OK."

Notes

- You can find a description of the various attributes by calling the online help via the "Motion Screen" or **F1** key, or by clicking on "Help".
- The "Properties" dialog box can be opened via the menu command **Edit > Object Properties**.

---

17 Printing Diagnostic Data

17.1 Printing Objects and Error Definitions

You can print out all data for one or more objects for an error definition.

The standard STEP 7 layout applies. Every page has a header and footer. The actual content consists of the preface and the corresponding units and error definitions.

- The printer outputs the numbers for blocks generated by S7-PDIAG in the preface,
- followed by selected units, motions, and error definitions.

Proceed as follows:

1. If you want to print a hardcopy of all generated data, select the top object from the unit overview and call menu command **Process Diagnostics > Print** in S7-PDIAG.
2. You can decide to print only the units or monitoring definitions selected from the unit overview of S7-PDIAG. Select an object (for example, a unit or a motion) or a specific error definition you want to print.
3. Call the menu command **Process Diagnostics > Print** or click the corresponding toolbar button to open the "Print" dialog box.
4. Set your print options.
5. Start printing with "OK".

Result:

The data for the object or error definition are printed on the default printer. If the hardcopy consists of more than one page, two periods are printed after the page number in the bottom right corner of the page. The last page does not have these periods, indicating that no more pages are to follow.

Notes

- You can set up the printer and the paper format (landscape or portrait)via the menu command **Process Diagnostics > Print Setup**.
- You can select the page layout for the printout (A4, A5, Letter, etc.) using the menu command **Process Diagnostics > Page Layout**.
- You can display the print preview via the menu command **Process Diagnostics > Print Preview**.
- You can set central headers and footers for your document in SIMATIC Manager.

17.2 Print Preview

You can open a print preview of specific objects (for example, a unit or a motion) before you actually start printing.

Proceed as follows:

1. Select the required print job.
2. Select the menu command **Process Diagnostics > Print Preview**. The image is then displayed in the print preview.

Note

If the hardcopy consists of more than one page, two periods are printed after the page number in the bottom right corner of the page. The last page does not have these periods, indicating that no more pages are to follow.

17.3 Page Layout

In S7-PDIAG, you can set up the page layout for printing objects (for example, a unit or a motion) or error definitions (A4, A5, Letter, Legal, etc.).

Proceed as follows:

1. Select the menu command **Process Diagnostics > Page Layout** to open the "Page Layout" dialog box.
2. Here, you specify the required page layout for your printout.
3. Confirm your entries with "OK."

Notes

- The print settings of the SIMATIC Manager are used in S7-PDIAG.
- Changes to the print settings in S7-PDIAG are not saved in the session memory.
- If you set a page format with margin (for example, A4 Margin), the hardcopy has a left margin that you can use to punch holes for binding.
- Call the menu command **Process Diagnostics > Print Preview** to check your settings.
- You can set global headers and footers for your document in SIMATIC Manager.

---

18 Exporting Diagnostic Data

18.1 S7-PDIAG Export Formats

After you have finalized your process diagnostics with S7-PDIAG, you can export the generated data for use by other applications. Two export formats are available:

- CSV and
- XML.

CSV format

You can open *.csv files in Microsoft Excel, for example. A semicolon is separates the listed elements. Diagnostic data are output in the specific language.

We have discontinued further development of the CSV format as of S7-PDIAG V5.0.

XML format

As of S7-PDIAG V5.1, the new XML format has been added to the old CSV format.

This XML format also contains new attributes and structure information. Attributes are always named in English language and their values are always output in all languages available.

Display of Export Data in CSV Format

Data exported by S7-PDIAG are always displayed in the same form:

- In the first section of the file, all S7-PDIAG objects assigned the corresponding attributes are listed as a comment, identified by "C:". Each object is assigned its own line.
- The second section of the file lists all user data for the above-mentioned S7-PDIAG objects and in the same form.

Example of CSV format display

C: Block container;Name;Author;Date when created;Time when created; etc.

C: Unit;Name;Author;Date when created;Time when created; etc.

C: Motion;Name;Author;Date when created;Time when created; etc.

C: Action monitoring;Name;Author;Date when created;Time when created; etc.

Block container;Blocks;;05.06.1998;07:36:42;10.06.1998;17:54:01;;

Unit;""""Drill"""";;05.06.1998;07:38:29;05.06.1998;07:55:00;;

Movement;""""Drill"".Clamp.m";;05.06.1998;07:38:29;;

Action monitoring;#m.Final_Position[0]:Action monitoring;;05.06.1998;07:51:54;08.06.1998;09:16:03;;;1;Yes(1);Target limit$$u$$ $$d1$$ not reached;;#m.Final_Position[0];FB100;Yes (1);Yes (1);Positive (1);#m.Control1;T#1M;

The diagnostic data are output language-specifically. Your export file contains the complete diagnostic data.

Advantages

- Using these exported data and the message log generated during the process, you can calculate down times and error frequency within your system, for example.
- Export data are also useful for determining the relationship between a specified message ID and the actual message.
- Other optional packages such as S7-GRAPH can also enter their messages into this export file.

18.2 Export of Diagnostic Data

The export includes all diagnostic data generated by S7-PDIAG in the selected format. All printable attributes of the entire program and all subordinate objects are exported.

Procedure:

1. Call menu command **Process Diagnostics > Export** to open the ”Export” dialog box.
2. Enter the name of the export file and format "*.csv" or *.xml". XML format is default.
3. Start the export with ”Save.”

The diagnostic data will now be exported to the selected ASCII file. You can generate those data using your own tool in order to create a basis for error statistics, for example.

19 Sample Project 'Drill Press'

19.1 Example of Monitoring Motions of a Drill Press

We shall use the example of a drill press below to show you an easy way of working with motion monitoring, by using an S7-PDIAG function block that contains the data interface to the OP as well as the monitoring definitions for S7-PDIAG.

Requirements

We shall presume the existence of the following software and hardware components for programming and debugging the "Drill press" sample project:

- PG/PC with STEP 7 standard software and optional S7-PDIAG software package
- MPI connection to the S7-300 or S7-400 PLC, with 16 digital inputs and 8 digital outputs,
- or alternatively to the PLC: The optional S7 software package "PLC Simulation"
- In order to familiarize yourself with the complete range of functions, you also require a display device with the process diagnostics package ProAgent.

19.2 Engineering Task and Function Diagram

A monitoring definition is to be programmed for the automation of a drill press. The structure of the drill press is predefined in a system diagram and the process is defined in the form of a function chart.

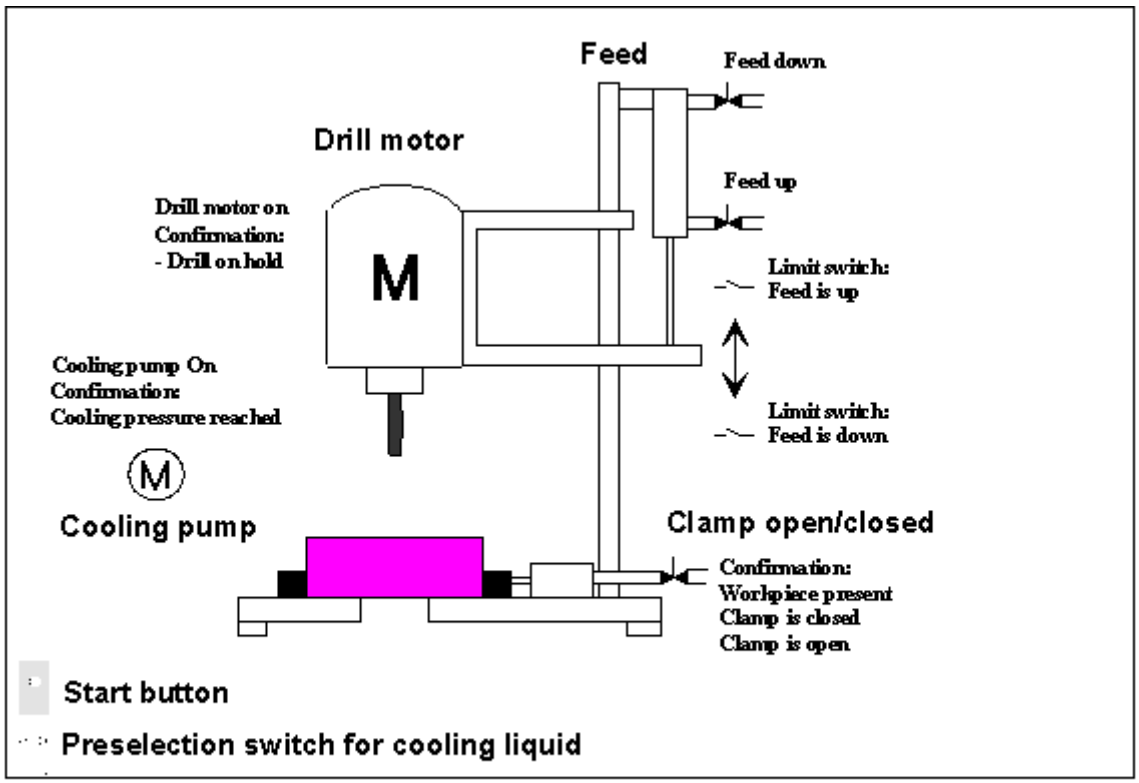
Structure of the Drill Press

- The drill press consists of the following elements:
- Drill drive motor with drill run/standstill feedback systems.
- Start button and coolant selector switch.
- Coolant pump with feedback "coolant pressure reached".
- Clamping system with feedback "tensioning setpoint reached".
- Vertical drill feed with up/down limit switch.

Initial state

The initial state of the drill is defined as follows:

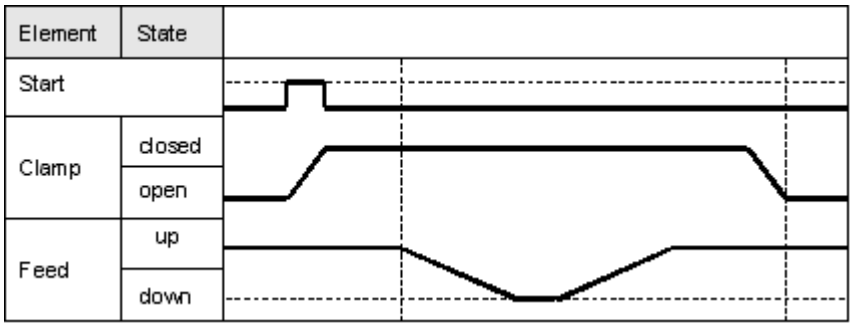
- Drill drive motor and coolant pump not in operation.
- Drill feed/drill in upper position.
- No workpiece mounted.



Function Diagram Drilling Sequence

Drill operation is split into the following steps:

- Inserting the workpiece (manually)
- Actuating the coolant selector switch, if required (depends on the material to be processed)
- Actuating the start button to start the machine (drive motor startup)
- Tensioning the workpiece in the chucking device according to the setpoint pressure
- Switching on the cooling pump (depends on the preset)
- Lowering the drill feed to lower target position (drilling)
- Dwell time at lower setpoint position of 0.5 s (drilling)
- Raising the drill feed to the upper target position
- Releasing the workpiece, switching off the drill motor and cooling pump
- Removing the workpiece (manually).



Determining Units

In this example there is exactly one system unit, namely the drill.

Since the drill does not have an operating mode selection switch, the UDT_S_Unit is used for this system unit.

Determining Motions

As shown in the diagram, there are two motions:

- Feed
- Clamp

Defining I/Os

List the corresponding I/Os for the drill press in a table.

If you decide to refer to symbolic programming, include the symbolic names for your absolute I/O addresses (e.g. Input I0.4 "ClampingTension_OK") and explanatory comments (e.g. "Message: Tensioning setpoint for workpiece not reached") in your list.

We shall presume in our example that the switches and contactors of the drill press will be controlled via the I/Os of the digital I/O module of an S7300 PLC. The I/O module used is equipped with 8 inputs and 8 outputs. The default I/O addresses of the module at slot 4 are: I0.0 to I0.7 and Q0.0 to Q0.7.

The table below lists the corresponding I/O for the drill press.

Absolute Addresses	Symbolic Addresses	Comment
Inputs	**In the program (E)**	
I0.0	Drill_motor_run	Feedback: Drill running at setpoint speed
I0.1	Drill_motor_Standstill	Feedback: Drill standstill
I0.2	Drill_Down	Limit switch for the bottom position of the drill
I0.3	Drill_Up	Limit switch for the upper drill position
I0.4	Clamping_Tension_ok	Feedback: workpiece clamping force reached setpoint
I0.5	Preselection_KLM	Coolant selector switch (depends on the workpiece)
I0.6	Coolant_Pressure_ok	Feedback: coolant pressure reached
I0.7	Start_Button	Start button of drill
Outputs	**In the program (A)**	
Q0.0	Drill_motor_On	Switch on drill motor
Q0.1	Coolant_pump_On	Switch on cooling pump (dependent on workpiece)
Q0.2	Drill_Lower	Lowering the drill feed to lower target position
Q0.3	Drill_Raise	Raising the drill feed to the upper final position
Q0.4	Workpiece_Clamp	Mounting the workpiece with setpoint clamping force

19.3 Program Structure of the Drill

The following table provides an overview of the blocks used in the user program and their functions:

Block	Name	Function
FB 1	Control drill	Control process of the drill
DB 1	Drill	Data (= process unit) of the drill
FB 100	Motion_to_Limit_Switch	FB template for motion with 2 final positions
OB 1	PLC cycle	Cyclic processing of the user program
OB 100	PLC startup	Starts up the user program
FB 44	Error_Detection FB	S7-PDIAG: Error recognition
DB 44	Error_Detection DB	S7-PDIAG: Error recognition
FB 45	Initial_Val_Acquisition FB	S7-PDIAG: Initial value acquisition
DB 45	Initial_Val_Acquisition DB	S7-PDIAG: Initial value acquisition
UDT 1	UDT_Unit	Data structure of a unit with operating modes
UDT 2	motion	Data structure of a motion
UDT 3	UDT_S_Unit	Data structure of a unit without operating modes

Model of Motion FB100

This block is a model of a motion FB that contains:

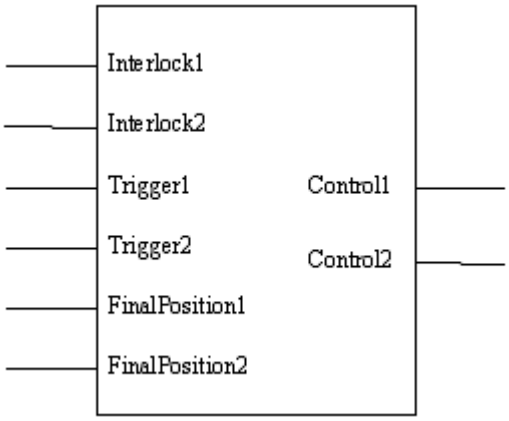
- The data interface between the display device and the user program, and
- The monitoring definitions (error definitions) for the motion.

The block simply needs to be interconnected in the actual user program for the drill. This makes motions much easier to handle.

The program comment for this block provides a detailed description of the individual networks, monitoring definitions, and their function.

Using FB100

Below you can see the calling interface of motion FB100:



Use of the Parameters

The following table shows how the parameters are used. The identifiers 1 and 2 represent the respective direction of the motion:

- 1= left at the display device (HMI)
- 2= right at the display device (HMI)

Interlock1/2	Interlocks which, if not fulfilled, prevent the motion from moving in the respective direction. Note: take the different operating modes into consideration. You can also insert the corresponding different network branches and interconnect them to this input.
Trigger1/2	Trigger which ensures that the motion can be moved in the required direction. Note: The motion block already contains the logic for operator control of a motion via the motion screens of the operator panel. This means that you only need to interconnect the trigger here, for example, for automatic operation.
FinalPosition1/2	Input showing that the motion has reached or left the respective final position. Note: If you set the option "Preset Final Positions" when you compile the program in S7-PDIAG, the system automatically enters the final position identifiers for the motion screen of the display device.
Control1/2	Triggers the outputs for the respective direction of the motion.

Monitoring Definitions in FB100

In our example, the following monitoring definitions have been configured in the motion FB100:

- **Interlock monitoring definitions** (Object should move, but is not allowed). With these monitoring definitions, initial value acquisition and therefore criteria analysis are activated. If an interlock error occurs, the criteria analysis on the display device shows which signals are missing in order for the motion to move. At the same time, the programming logic with which the interlock inputs of this function block are interconnected is analyzed.
- **Action monitoring definitions** (Object is moving, but does not reach the target position within a specified time).
  Initial value acquisition will not be enabled for this monitoring definition, since only the time-to-go for the object to reach the target position is being monitored. A default time of 1 minute has been configured here. If the requirements cease to apply while the motion is in progress, and the motion is allowed to continue, interlock monitoring is triggered.

Controlling the Drill in FB1

FB1 contains the control program for the drill press.

In the first step, you declare a "Drill" unit in the code section of this data area of this FB. Since the "Drill" unit does not recognize different operating modes, the UDT_S_Unit is used as identifier. You can use the group error bit in this UDT to determine whether or not there is an error in this unit.

The individual motions are then specified as a multiple instance for this unit. This means that you do not require a separate data block for each individual motion; instead, all the data for the motions of this unit can be combined in one block.

The drilling sequence is controlled in the program section. The motion FB is used twice for this function. S7-PDIAG recognizes this multiple use and sets the appropriate monitoring definition for each motion. The drill press FB itself is not assigned any parameters and is called in the user program cycle (OB1).

Drill Press DB1

This data block contains all the data required to control the drill.

When you call FB1 in OB1 to control the drill, you are automatically asked whether the corresponding instance data block has been generated. The symbolic name of this data block is accepted by the display devices as the name of the unit.

S7-PDIAG Monitoring Blocks

S7-PDIAG generates the monitoring blocks FB 44/45 and DB 44/45. They contain both the logic for error detection and the information required for initial value acquisition. It suffices to call error recognition FB 44 and its corresponding instance data block DB 44 at the end of the user program cycle (OB1).

19.4 Working with the Example

Preparations

In order for the example to run correctly, you require input bytes 0 and 1 and output 0 in your controller. First switch the controller to STOP.

- If you are working without a display device, you can display the messages generated by S7-PDIAG on your PG/PC via the menu command **PLC > CPU Messages**. In this case, enter a check mark in column "A" and select "Foreground".
  In this case, however, you cannot display the unit overview, the motion screen, and the criteria analysis. Instead, you can open FB1 and monitor it online.
- If you are working with a display device, create a new project and insert your CPU and the relevant display device.
- Doubleclick the network symbol to check that both the CPU and the display device are connected to a common network. Then copy the sample program into the program below your CPU.
- Download the sample program to the CPU and, if necessary, to the configuration for the display device.

Procedure

The error messages that occur will guide you through the example.

You will see how S7-PDIAG and ProAgent help you to "Run" the drill and offer you support when errors occur.

Now select the message screen on the display device, or disable the function "CPU Messages".

How to set the initial state

In the first step, set all inputs to "0" and switch the CPU to "RUN".

- This message will be output:
  "Open clamp interlock".

This indicates that the clamp is closed and that you should run the drill towards "UP" direction while an approach towards this direction is not permitted.

Press the "Criteria analysis" key on the display device and monitor the tensioning motion network.

- The criteria analysis returns the following result:
  A I1.0 DrillStandstill Feedback: Drill not rotating

This is the missing signal that allows you to run the clamp towards "Up" direction. Now set input I0.1 "DrillStandstill" to "1".

The system reports the outgoing error message, indicates that output Q0.5 "ClampUp" is set and that the clamp will be driven towards "Up" direction subsequently.

- This message is output after one minute:
  "ClampUp: Final position not reached:"

This means that the final position showing the open clamp was not reached.

You will see that the trigger for the motion is flashing in the motion screen, but the final position has not yet been reached. Now set input I1.1 Is_Open" to 1." The message disappears and the motion screen displays the final position reached.

The initial settings of the drill are now complete.

Drilling Sequence

How you can "simulate" the complete drill sequence.

Set input I0.7 "Start_Button" to On."

- This message will be output:
  "Close clamp interlock".
- The criteria analyse returns the following result:
  A I0.1 WorkpieceDa Feedback: Workpiece present in clamp

As there is no workpiece in the clamping device, clamping cannot take place. Simulate the insertion of a workpiece by setting input I1.0 "Workpiece_Present".

The clamp now closes. The system sets output Q0.4 "Is_Closed" and at the same time switches on the drill by setting Q0.0 "Drill_Motor_On".

- After the specified time of one minute has expired, the following error message appears:
  "*Clamp has not reached final position".*

Now first reset the final position I1.1 "Is_Up" and set the final position I1.2 "Is_Closed".

- Now this message will be output:
  "Interlock drill feed down".
- The criteria analysis returns the following result:
  AN I0.1 DrillStandstill Feedback: Drill not rotating

Now simulate rotary motion of the drill motor by resetting input I0.1 "Drill_Standstill". Please note that a negative logic will be displayed due to the "UN" identifier.

The drill feed now moves down and drilling takes place. Output Q0.2 "Drill_Feed_Down" indicates this motion.

- After the 1-minute monitoring time, this message is output:
  "Drill feed down: Final position not reached".

First reset the final position I0.3 "Is_Up" and then set the final position I0.2 "Is_Down".

Now the drill feed moves "Up" again. Output Q0.3 "Drill_Feed_Up" indicates this motion.

- After the 1-minute monitoring time, this message is output:
  "Drill feed up: Final position not reached".

First reset the final position I0.2 "Is_Down" and then set the final position I0.3 "Is_Up".

Finally all that is left to do is to open the clamp again.

- However, you are first shown this message:
  "Open clamp interlock".
- The criteria analysis returns the following result:
  A I0.1 DrillStandstill Feedback: Drill not rotating

Now acknowledge this message again.

The system now sets output Q0.5: "Release_Clamp".

- After the 1-minute monitoring time, this message is output:
  "Open clamp: Final position not reached".

Now first reset the final position I1.2 "Is_Closed" and set the final position I1.1 "Is_Open".

The drilling is now completed and you can remove the workpiece by resetting input I1.0 "WorkpieceDa". On the motion screen you can now see that the left triangle of the "Clamping" motion is no longer filled, indicating that this motion can now no longer be executed. As soon as you insert a workpiece, the clamp can be used again.

Further Drilling Processes

You can now "simulate" the same sequence and include the coolant functions.

Summary

With a relatively small time spent for programming under S7-PDIAG and ProAgent you will receive automatically generated motion screens and detailed error information. This enables you to reduce the down times of your automation solution and contributes to increased productivity.

20 Glossary

Action Monitoring

Action monitoring is one of the four motion monitoring modes. In this mode, the function checks whether the target final position) has been reached within a certain time (Action time) after a control command was output (Trigger).

Action Time

The action time is the time within which a motion must be completed.

Actual Final Position

The actual final position represents the current position of the motion. When using UDT_Motion, this is defined as "Final_Position(s)".

Address Monitoring

The address monitoring function is directly interconnected with an address known as the initial diagnostic address.

The address monitoring function can be used to detect whether the initial diagnostic address has reached a defined level on expiration of a certain (monitoring) time. If this is the case, the error is registered as an incoming error, and as outgoing error when the address level changes again.

Depending on whether you select level or edge monitoring, the delay time is triggered either instantaneously, or by the next active edge.

Associated Value (Definition)

An associated value (or address) can be ”added" to a message text. This value is acquired by S7-PDIAG at the time an error is detected. The associated value is displayed on the HMI, in the message text at the configured location. To do this, enter the corresponding formal address in the message text.

Permissible Addresses for an Associated Value:

An associated value can be an address of the type BOOL, BYTE, CHAR, WORD , INT, DWORD, DINT and REAL of the I, Q, M or DB area. You can also enter local tags of an function block which has an error definition as associated values. Depending on the type-instance concept, these values are expanded during compilation.

You can define the position and format of the associated value in the message text by creating a description block for the associated value in the following syntax:

@1<element type><format definition>@

The associated value is inserted at the description block position in the message text.

Element type of associated values:

Assigns a definite data type to an associated value.

Element type	Data type
X	BOOL, BYTE, WORD, DWORD, INT, DINT
C	CHAR
R	REAL

Format definition of associated values:

You can select on of the following characters to define the output format of an associated value. The format definition starts has a "%" prefix.

Format definition:	Description:	Display range:
%d	Decimal with preceding sign	-2147483648.. +2147483647
%u	Decimal without preceding sign	0..4294967295
%X	Hexadecimal	0..FFFFFFFF
%b	Binary	11111111111111111111111111111111
%[i]X	Hexadecimal number with i digits	
%[i]u	Unsigned decimal with i digits	
%[i]d	Signed decimal with i digits	
%[i]b	Binary number with i digits	
%[i][.y]f	REAL number	Signed value with the format [-]dddd.dddd dddd: one or several numbers with "y" digits after the decimal point and a total number of digits "i".
%1s	CHARacter	ANSI characters

Examples of the description block for an associated value:

@1X%6d@: Display associated values in DEC format with max. 6 digits

@1X%1b@: Display associated values in BOOL format with "0" or "1"

Example of a description block for CHAR or REAL data types:

@1C%1s@: Display associated value "A" in CHAR format

@1R%4.2f@: Display associated value "5.4" as "5.40" in REAL format

Note

To ensure that the associated values are included during compilation, the "Include associated values" check box in the "Compile" tab ("Customize" dialog box) must be selected.
To open this dialog box, select the **Options > Customize > Compile** menu command.

Auxiliary Networks

Auxiliary networks generated by S7-PDIAG describe the auxiliary logic you are using. This auxiliary logic is used to analyze a relevant network. S7-PDIAG uses auxiliary networks for criteria analysis.

Criteria Analysis (Definition)

Criteria analysis determines the cause of error based on the logic defined in the user program, is performed on the HMI, and used to trace the cause of error. Criteria analysis starts at the initial diagnostic address, and evaluates the initial values of all networks which determine the value of the initial diagnostic address.

Positive criteria analysis assumes signal state "1" of the initial diagnostic address being correct, while negative criteria analysis assumes signal state "0" being correct.

To be able to perform a criteria analysis on the display device, initial-value acquisition must be set for this monitoring function.

Note

- Subnets will be ignored when exclusion addresses are defined
- At FCs, criteria analysis can not track data beyond block boundaries. Call parameters are not transferred to FCs blocks using static data, as in the case of FBs and corresponding instances. It is thus **not** possible to access those data at a later time, or track those parameters beyond block boundaries in a criteria analysis.

Details

Opens a window for the display of error messages and address information, for editing symbols, controlling addresses, comparing blocks, and for editing error definitions for process diagnostics.

Error Definition

An error definition provides a precise description of the error in the monitored process. Error definitions can be appended to addresses in the LAD/STL/FBD Editor which contain an assignment, a set or reset instruction, the FB and FC calls.

Formal Addresses Substituted During Compilation

S7-PDIAG substitutes the following formal addresses during compilation:

$$u$$, $$u1$$	Name of the higher-level unit
$$u2$$ to $$u9$$	The name of the corresponding higher-ranking unit. **Example:** You create an error definition with the formal address $$u3$$ in the fifth subunit of the S7-PDIAG tree structure. The unit value is therefore "u" or "u1". Starting at this value, the counter increments until the higher-ranking unit u3 is reached. Unit u3 is then output to the HMI.
$$ur$$	Name of the top unit within the tree structure
$$m$$	Name of the motion Important note:
$$o$$	Diagnostic entry address of the instance error definition (symbolic format if no symbol exists in the absolute format)
$$d1$$	Name of the motion direction 1
$$d2$$	Name of the motion direction 2
$$a$$	Diagnostic entry address in absolute format
$$s$$	Diagnostic entry address in symbolic format
$$c$$	Symbol comment for the diagnostic entry addresses **Note:** In the message text you can replace the formal address with the symbol comment from the symbol table.
$$CpuName$$	Name of the configured CPU. If no CPU was configured, the formal address will be retained and is not substituted during compilation.

You can select the name source. Available are:

- The names of units
- The motion name
- The direction texts
- The name of the instance error definition of the initial diagnostic address

Please note:

- Formal addresses are replaced in the specific language used.
- The last component of the name is always used for units and motions. This means that the name is composed of the element located between the end and the first dot on the way to the beginning of the name.

Example:

Name	Result
Drilling_Machine.Feed	Feed
Drilling_Machine.Drilling_Machine.Feed	Drilling_Machine.Feed

Note

More than one formal address within the opening and closing characters ($$) is **not** permitted.

Formal Addresses Replaced in the Message Text

The system replaces the following addresses when displaying messages on the HMI:

You can define the position and format of an associated value in the message text by creating a description block for this value. Use this syntax:

- @1<element type><format definition>@

The associated value replaces this description block at this position in the message text.

Permissible Addresses for an Associated Value:

An associated value can be an address parameter of the type CHAR, WORD , INT, DWORD, DINT and REAL from the I, Q, M or DB area. You may also enter local tags as associated values in a function block that has an error definition. Depending on the type-instance concept, those values are expanded during compilation.

You may define the position and display format of the associated value for the message text.

Element type of associated values:

Sets a data type definition for an associated value.

Element type	Data type
X	BOOL, BYTE, WORD, DWORD, INT, DINT
C	CHAR
R	REAL

Format Types for Associated Values:

You can select one of the characters shown below to define the output format of an associated value. The format definition has a "%" prefix.

Format definition:	Description:	Display range:
%d	Decimal with preceding sign	-2147483648.. +2147483647
%u	Decimal without preceding sign	0..4294967295
%X	Hexadecimal	0..FFFFFFFF
%b	Binary	11111111111111111111111111111111
%[i]X	Hexadecimal number with i digits	
%[i]u	Unsigned decimal with i digits	
%[i]d	Signed decimal with i digits	
%[i]b	Binary number with i digits	
%[i][.y]f	REAL number	Signed value in the format [-]dddd.dddd dddd: one or several digits with "y" after the decimal point and a total number of digits "i".
%1s	Character	ANSI character

Examples of the description block for an associated value:

@1X%6d@: Display associated values in DEC format with max. 6 digits.

@1X%1b@: Display associated values in BOOL format with "0" or "1".

Example of a description block for CHAR or REAL data types:

@1C%1s@: Display associated value "A" in CHAR format.

@1R%4.2f@: Display associated value "5.4" as "5.40" in REAL format.

Note

To ensure the inclusion of associated values in your compilation, set the "Include associated values" check box on the "Compile" tab ("Customize" dialog box.).
To open this dialog box, select the **Options > Customize > Compile** command.

Error Definition Priority

Error definitions can by assign priority classes 1 to 16. Each priority class is assigned one bit in a flag word you can set in the **Settings > Default** dialog box. This bit will be set when an error of the corresponding priority class occurs, and is not reset until all errors of this priority class have been cleared.

The priority class is weighted in analog to the HMI. Priority classes are assigned to the bits in the flag word as follows:

- Priority class 1: = bit 0 in the low byte of the flag word
- Priority class 16: = bit 7 in the high byte of the flag word

Note

All monitoring functions of projects created in S7-PDIAG Version 3.0 are implicitly assigned priority class "1."

Exclusion Addresses (Definition)

Exclusion addresses are addresses you define as "never causing an error". The criteria analysis function hides those addresses and the auxiliary networks in which they are located, if they have been assigned a "0" value (only in conjunction with ProAgent, Version 5.0 or higher).

You can define exclusion addresses in S7-PDIAG or in the "LAD/STL/FBD" Editor.

Executability

Executability means the execution of the motion is enabled. It is defined as "Executability1/Executability2" when using UDT_Motion.

Generating Instances

A function block may be assigned one or several instance data blocks. Each error definition at the FB requires one error definition per instance DB in order to monitor all of the instances.

The error definition configured only once in the function block and applies to all instances, i.e. by their generation, the instances are assigned the necessary instance-specific error definitions. You can then assign instance-specific values (for example, with different delay times) to those error definitions.

If you edit the error definition of an instance, this error definition will not be overwritten at the FB in the next compilation.

You should observe three groups here. When you edit a value in one of the groups outlined below in the instance, the group type is no longer updated.

- **Group 1:** name, author, comment
- **Group 2:** message-specific information, including the message text
- **Group 3:** monitoring logic, time, criteria analysis and actual value acquisition

If you delete the error definition of an instance, the error definition is created again on compilation.

Formal parameters are replaced by the relevant actual parameters when the instances are generated.

Group Error

A group error message is routed from an error-triggering subunit to the highest unit in the unit overview of S7-PDIAG, and is displayed on the HMI.

Global Monitoring (Definition)

With global monitoring you can define your own monitoring logic as a sequence of logical expressions. You create monitoring logic using the available language elements, i.e. you can create complex error monitoring definitions. The error state occurs when the defined conditions have been fulfilled.

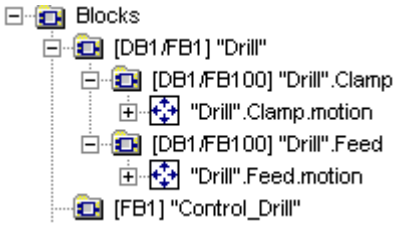
Hotkeys

The hotkeys are assigned directly to the digital inputs of the controller (for example, as hardware wiring or via a DP interface) via the digital outputs of the HMI. They allow immediate and direct operator intervention in the motion on the motion screen of the HMI.

Important Note

The motion name will be replaced in the message text by the formal address $$m$$. However, if you work with standard blocks, the name of the standard block replaces the motion name, but **not** the name of the actual UDT_Motion. In this case, use the formal address "$$u$$" in the message text.

Example:



In the example, FB100 represents a motion block in which the UDT_Motion has been implemented as local instance type named "motion".

Assuming you are going to use formal address $$m$$ in the message text of the monitoring functions in FB 100, this would lead in all applications to the name "motion." In this case, you should use "$$u$$", for this would then lead correctly to "Clamp" or "Feed".

Initial Diagnostic Address

The initial diagnostic address is the address to which the error definition in the block is logically linked. Error tracking starts at this address when a criteria analysis is to be performed. This address must either be an instruction, or a "Set" or "Reset" operation.

Initial Value Acquisition

If initial value acquisition for an error definition is set, all binary states of the addresses that were used to form the monitored address (initial diagnostic address) are recorded in the cycle in which an error is detected. You can use these to perform a criteria analysis, which will simplify troubleshooting.

Interlock Monitoring

Interlock monitoring is one of the four types of motion monitoring. The function checks whether the interlock condition (Executability) has been met after a motion control command Trigger) was output and a certain time (interlock time) has expired.

Interlock Time

The interlock time is the time within which the interlock conditions must be fulfilled.

Ladder Networks for Motion Programming

S7-PDIAG is supplied with default LAD networks for UDT_Motion functions in FB100 of the "ZEn06_01_S7PDIAG_drill" project which you can use to program motions:

- For one motion direction, using the hotkeys on the display device, and
- For one motion direction, without using the hotkeys, controlled manual mode or status display on the display device.

In addition to a convenient means of programming motions, those networks also take advantage of the "manual mode" function to allow motion control and debugging directly on a motion screen of the display device, for example.

Modifying Times

If an existing monitoring definition contains a monitoring time unequal to 0, you can modify it using the "Change Times" function, without having to create new monitoring blocks. The function is available in online and offline mode, and lets you determine the suitable monitoring time step- by-step.

Monitoring Blocks

Monitoring blocks are used for error detection, and for the acquisition of initial values and status data generated based on the error definitions created in S7-PDIAG. After their downloaded to the user program, the monitoring blocks execute process diagnostics functions.

Monitoring Definition

S7-PDIAG supports various process monitoring functions you can create using the error definitions.

These include:

- Address monitoring
- Global monitoring
- Action Monitoring
- Startup Monitoring
- Reaction monitoring and
- Interlock monitoring

Monitoring Logic

The monitoring logic is defined using the language elements of S7-PDIAG, and is used to monitor your process.

S7-PDIAG provides monitoring logic templates for address and motion monitoring which you can adapt to suit your requirements.

For general monitoring functions, you can program your own monitoring logic using the language elements provided in S7-PDIAG.

Motion (Definition)

A motion represents a process sequence, for example a die moving up and down. Motions can be monitored by means of error definitions.

When using UDT_Motion and the LAD networks for motion programming included with S7-PDIAG, you can visualize and control motions without the need of further programming on the motion screens of the HMI.

Motion Monitoring (Definition)

Motion monitoring functions are specifically tuned to monitor motions within the process. Monitoring logic entries are based on a default logic which you can adapt to suit requirements. The error state occurs when the defined conditions have been fulfilled.

You can select any of the following default motion monitoring functions:

- S7-PDIAG: Action Monitoring Definitions
- S7-PDIAG: Startup Monitoring Definitions
- S7-PDIAG: Reaction Monitoring Definitions
- S7-PDIAG: Interlock Monitoring Definitions

Motion Monitoring Logic

The action monitoring logic is defined as follows:

ONDT (<trigger>,<action time>)
AND
NOT <target final position>

Note

In this case, the initial diagnostic address represents the final position.

The startup monitoring logic is defined as follows:

ONDT (<trigger>,<startup time>)
AND
<actual final position>

Note

In this case, the initial diagnostic address represents the actual final position.

The reaction monitoring logic is defined as follows:

ONDT (<position flag> AND NOT <target final position>,<reaction time>)

Note

In this case, the initial diagnostic address represents the target final position.

The interlock monitoring logic is defined as follows:

ONDT (<trigger>,<interlock time>
AND
NOT<executability>

Note

In this case, the initial diagnostic address represents the executable function.

Monitoring type

Refers to an error type definition, in contrast to its instance.

Other Error Definitions

This term refers to error definitions in S7-GRAPH and S7-HiGraph.

Position Flag

A "Position_Flag(n)" position flag is defined when UDT_Motion is being used.

Reaction Monitoring

Reaction monitoring is one of four types of motion monitoring, and is used to monitor whether a final position is reached within a certain time (Reaction Time) after passing a position flag, or whether an object has moved out of the final position by more than the defined time (Reaction Time).

Reaction Time

Represents the time limit to reach the final position, or the maximum time during which an object may leave the final position.

S7-GRAPH

The S7-GRAPH programming language enhances STEP 7 functionality with graphic functions for programming sequential control systems.

The S7-GRAPH programming language is a comfortable means of describing sequential control systems (sequencer programming). The process sequence is here split into several steps which in particular contain actions for controlling outputs. The transition to the next step is determined by transitional conditions.

S7-GRAPH is an optional package of STEP 7.

S7-HiGraph

The S7-HiGraph programming language for S7-300 and S7-400 systems enhances STEP 7 functionality with a system for programming status graphs.

It is a comfortable means of creating non-sequential, asynchronous actions by means of status diagrams. The system is here organized by function units which may assume different states and are capable of automatic synchronization by means of data exchange.

S7-GRAPH is an optional package for STEP 7.

Standard Group

The standard group is created by default. From this group, units can be moved to other user-defined groups (max. 15.)

When you delete a user-defined group, all its units will be automatically returned to the standard group.

Startup Monitoring

Startup monitoring is one of the four motion monitoring modes, used to check whether an object moves out of the actual final position (Actual final position) within a certain time (Startup time) after a control command (Trigger) was output.

Startup Time

The startup time is the time within which a motion must have been started.

Status

In S7-HiGraph, a status represents an element of a status graph and defines a logical, mechanical or time-based "state." Each state can be assigned actions, and is displayed as a circle in S7-HiGraph.

Step

In S7-GRAPH, PLC jobs are divided into separate steps. The steps describe the actions the PLC performs at a defined plant state.

Target Final Position

The target final position represents the position the current motion should reach. It is defined as "Final_Position(s)" when using UDT_Motion.

Templates

In addition to the default monitoring functions supplied with S7-PDIAG, you can also create user-specific templates for specific monitoring functions. The templates facilitate project engineering and save time and effort. You can also save incomplete monitoring logic data to user-defined templates.

Note: only one message class will be created per template, which is why a message number will not be assigned in a template. You can nevertheless configure the message text for your monitoring function in the template.

Termination Addresses (Definition)

When a termination address is reached, the use of auxiliary networks for criteria analysis will be canceled. You can define termination addresses in a list.

Transition

A transition in S7-GRAPH represents an element of a sequencer which contains the transitional step conditions.

In S7-HiGraph, a transition represents the status graph element which contains the transitional conditions.

Trigger

The trigger initiates the motion. When using the UDT_Motions, it is defined for interlock monitoring as "Trigger1/Trigger2", and as "Control1/Control2" for startup and action monitoring.

UDT (User Data Types)

The User Data Types supplied with S7-PDIAG can be saved as blocks which may be reused in multiple instances:

- as "standard" data type,
- as template for generating blocks of the same data structure.

UDTs available in S7-PDIAG:

- UDT_Unit
- UDT_S_Unit
- UDT_Motion

UDT_Motion

UDT_Motion represents a standardized interface between S7-PDIAG and the HMI. It contains all parameters required

- to display motions on the motion screens of the HMI, without the need of additional programming, and
- to control of those motions on the motion screen of the HMI.

Note

Prerequisite for motion programming is the use of the LAD Networks for Motion Programming in FB 100 of the sample project "ZEn06_01_S7PDIAG_drill" which is supplied in the S7-PDIAG software package.

UDT_Unit

The UDT_Unit contains the information required by the HMI to associate an error alarm with the faulty program location. The UDT_Unit contains defines:

- group error detection and group error acknowledgement
- 16 modes, with two default definitions of "Manual" and "Auto". You can define the remaining 14 operating modes to suit your requirements.

UDT_S_Unit

UDT_S_Unit only contains the group error address and the group error acknowledgement.

This saves memory space, and the process unit mode definition does not need to be "looped through" to all subunits.

Unit

Units are used to structure the process view based on technically interrelated components. In a project configuration where each block controls a physical object in the process (for example, a press, a stamp, or a safety guard), the units represent an image of your process. A unit exists for each block with diagnostics function in the user program.

Units can also store data shared by all other units, motions, and function blocks of the lower hierarchy layers.

A unit may contain error definitions, motions, and nested subunits.

This lets you group single errors and motions in a single technological unit, and facilitates locating of process errors.

Units are represented alongside with other objects in a tree structure in the unit overview. Units for a data block, function, or organization block are also visible in the unit overview on the HMIs.

Unit Overview

The unit overview represents the user interface of S7-PDIAG. The left pane of the unit overview displays a tree structure of objects such as units and motions. The right pane displays the monitoring functions and nested units which have been created for the object selected on the left pane. The output window provides information and error messages and lets you declare tags and block parameters.

User Block

A user block is a function block with a defined interface such as contained in the FB 46 supplied with the sample program. Users my program these to define specific reactions to process errors. The user block is always called when S7-PDIAG reports an "incoming" or "outgoing" error.

Index

F

G

H

I

L

M

N

O

P

R

S

T

U

W

X