SIEMENS

SIMATIC

S7-GRAPH V5.3 for S7-300/400 Programming Sequential Control Systems

Manual

Edition 02/2004
A5E00290656-01

Safety Guidelines

This manual contains notices intended to ensure personal safety, as well as to protect the products and connected equipment against damage. These notices are highlighted by the symbols shown below and graded according to severity by the following texts:

Danger
indicates that death, severe personal injury or substantial property damage will result if proper precautions are not taken.

Warning
indicates that death, severe personal injury or substantial property damage can result if proper precautions are not taken.

Caution
indicates that minor personal injury can result if proper precautions are not taken.

Caution
indicates that property damage can result if proper precautions are not taken.

Notice
draws your attention to particularly important information on the product, handling the product, or to a particular part of the documentation.

Qualified Personnel

Only **qualified personnel** should be allowed to install and work on this equipment. Qualified persons are defined as persons who are authorized to commission, to ground and to tag circuits, equipment, and systems in accordance with established safety practices and standards.

Correct 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.

This product can only function correctly and safely if it is transported, stored, set up, and installed correctly, and operated and maintained as recommended.

Trademarks

SIMATIC®, SIMATIC HMI® and SIMATIC NET® are registered trademarks of SIEMENS AG.

Third parties using for their own purposes any other names in this document which refer to trademarks might infringe upon the rights of the trademark owners.

Siemens AG
Bereich Automation and Drives
Geschaeftsgebiet Industrial Automation Systems
Postfach 4848, D- 90327 Nuernberg

Disclaimer of Liability

We have checked the contents of this manual for agreement with the hardware and software described. Since deviations cannot be precluded entirely, we cannot guarantee full agreement. However, the data in this manual are reviewed regularly and any necessary corrections included in subsequent editions. Suggestions for improvement are welcomed.

Preface

Purpose of the Manual

This manual provides you with a complete overview of programming with S7-GRAPH. It supports you during the installation and setting up of the software. It includes explanations of how to create a program, the structure of user programs, and the individual language elements.

The manual is intended for persons with the appropriate qualifications who are involved in programming, configuration, commissioning, and service of programmable logic controllers.

We recommend that you familiarize yourself with the example in Chapter 3 "Designing a Sequential Control System Based on the Example of a Drill". This will help you to get to know S7-GRAPH quickly.

Required Experience

To understand the manual, you should have general experience of automation engineering.

In addition to, computer literacy and the knowledge of other working equipment similar to the PC (e.g. programming devices) under the operating systems MS Windows 2000 Professional or MS Windows XP Professional are required. Since S7-GRAPH is a component of the STEP 7 basic software, you should have knowledge of the basic software conveyed in the "Programming with STEP 7 V5.x" manual.

Scope of the Manual

The manual is valid for the S7-GRAPH V5.3 software package.

Documentation Packages for S7-GRAPH and the STEP 7 Standard Software

The following table provides you with an overview of the STEP 7 and S7-GRAPH documentation:

Manuals	Purpose	Order Number
S7-GRAPH V5.3 for S7-300/400	Basic and reference information explaining how to create a program, the structure of user programs and the individual language elements.	The manual cannot be ordered separately. It is available on the product CD, the manual collection and on the internet.
Basics of STEP 7: • Getting Started and Exercises with STEP 7 V5.3 • Programming with STEP 7 V5.3 • Configuring Hardware and Connections with STEP 7 V5.3 • Converting from S5 to S7	The basics for technical personnel describing how to implement control tasks with STEP 7 and S7-300/400.	6ES7810-4CA07-8BW0
STEP 7 reference: • LAD/FBD/STL manuals for S7-300/400 • Standard and System Functions for S7-300/400	Reference work describing the LAD, FBD and STL programming languages as well as standard and system functions as a supplement to the STEP 7 basics.	6ES7810-4CA07-8BW1

Manual and Online Help

This manual is an extract from the Online Help. The manual describes the basic knowledge you require to work with S7-GRAPH. The online help contains detailed, step-by-step instructions and reference sections. Since the manual and online help have the same structure, you can change easily between manual and online help.

Access to Online Help

The help system is integrated in the software with several interfaces:

- The **Help** menu provides numerous menu commands: **Contents** opens the contents of the S7-GRAPH help system. **Introduction** provides an overview of programming with S7-GRAPH. **Using Help** provides detailed instructions on working with the online help system.
- The context-sensitive help system provides information about the current context, for example help on an open dialog box or active window. This can be displayed by clicking the "Help" button or pressing the F1 key.
- The status bar is another form of context-sensitive help. A brief explanation of each menu command is displayed here when you position the mouse pointer on a menu command.
- A brief explanation of the buttons in the toolbar is also displayed if you position the mouse pointer briefly over a button.

If you prefer to have a printout of the information in the online help system, you can print individual topics, books or the entire help system.

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

Training Centers

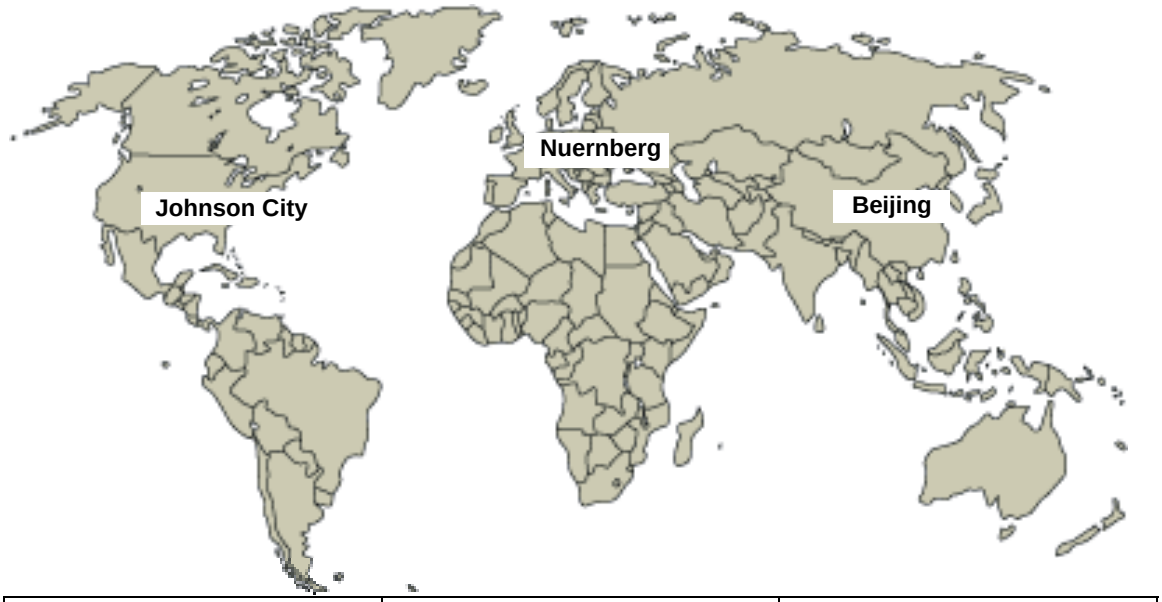
Siemens offers a number of training courses to familiarize you with the SIMATIC S7 automation system. Please contact your regional training center or our central training center in D 90327 Nuremberg, Germany for details:

Telephone: +49 (911) 895-3200.

Internet: http://www.sitrain.com

A&D Technical Support

Worldwide, available 24 hours a day:



Worldwide (Nuernberg) **Technical Support** 24 hours a day, 365 days a year Phone: +49 (180) 5050-222 Fax: +49 (180) 5050-223 E-Mail: adsupport@ siemens.com GMT: +1:00		
Europe / Africa (Nuernberg) **Authorization** Local time: Mon.-Fri. 8:00 to 5:00 PM Phone: +49 (180) 5050-222 Fax: +49 (180) 5050-223 E-Mail: adsupport@ siemens.com GMT: +1:00	**United States** (Johnson City) **Technical Support and Authorization** Local time: Mon.-Fri. 8:00 to 5:00 PM Phone: +1 (423) 262 2522 Fax: +1 (423) 262 2289 E-Mail: simatic.hotline@ sea.siemens.com GMT: -5:00	**Asia / Australia** (Beijing) **Technical Support and Authorization** Local time: Mon.-Fri. 8:00 to 5:00 PM Phone: +86 10 64 75 75 75 Fax: +86 10 64 74 74 74 E-Mail: adsupport.asia@ siemens.com GMT: +8:00
The languages of the SIMATIC Hotlines and the authorization hotline are generally German and English.		

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

Programming Language S7-GRAPH

The S7-GRAPH programming language enhances the functional scope of STEP 7 with a graphical programming interface for sequential controls.

S7-GRAPH allows you quick and clear programming of sequential operations you want to control with a SIMATIC PLC. The process is here split into single steps to provide a clear overview of the functional scope. The graphical display of the sequencer can be documented with pictures and text.

Actions to be executed are determined in these steps. Transitions control the transition between steps (conditions for switching to the next step). These conditions are defined with the help of the programming languages LAD (ladder logic) or FBD (function block diagram).

PLCopen Basis Level

S7-GRAPH as of V5.1 complies with the PLCopen Basis Level for sequential control systems as stipulated in the DIN EN 61131-3 standard.

If you have a sequential control system that conforms with the standard, you can now import it into the STEP 7 data management as an ASCII file using the SIMATIC Manager and then edit it in S7-GRAPH.

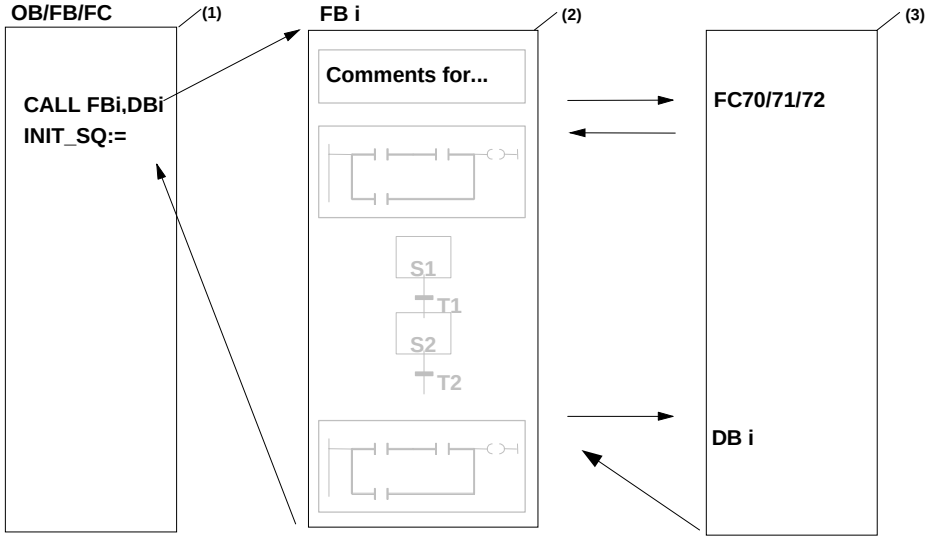
To create sequential control systems complying with the standard, select the "Application Settings" dialog box and activate the "IEC-compliant" in the "General" tab.

Sequential Control Blocks

A sequential control controls the process in a preset order that depends on certain conditions.

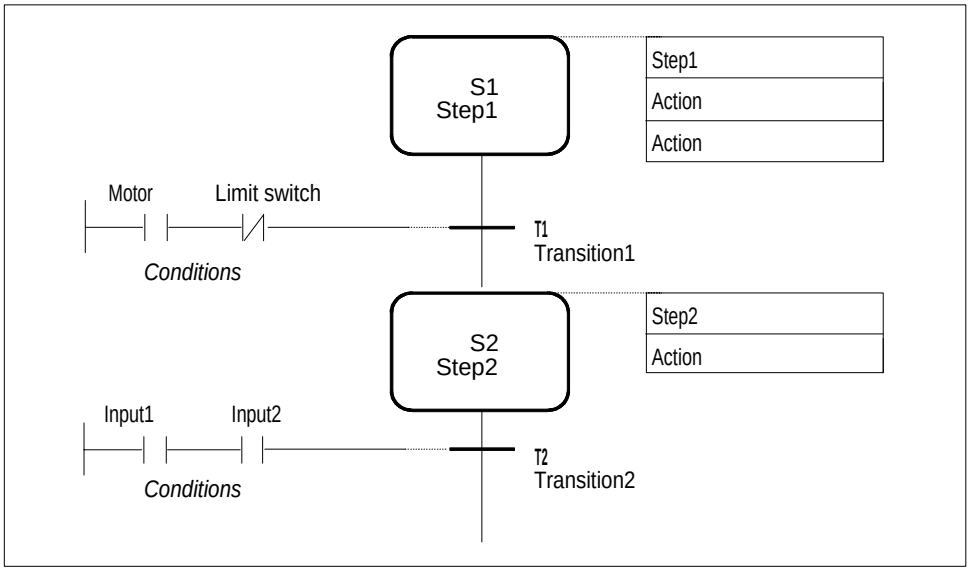
The complexity of the sequential control depends on the automation task. However, even the most basic system requires at least these three following blocks:

- A STEP 7 block (1) in which the S7-GRAPH FB (function block) is called. This block can be an organization block (OB), a function (FC), or another FB.
- An S7-GRAPH FB (2) that describes the individual subtasks and interdependencies of the sequential control system. These subtasks (steps) and interdependencies (transitions) can be organized in one or more sequencers.
- An instance DB (3) that contains data and parameters of the sequential control system. The instance DB is assigned to the S7-GRAPH FB and can be created automatically by the system.



Extract of a Sequencer with Steps and Transitions

You program sequencers in the S7-GRAPH FB. A sequencer consists of a sequence of steps that are executed in a fixed order depending on transitions (conditions for switching to the next step).



1.1 What's New?

Version 5.3 of the S7-GRAPH programming software has the following extended functions and modifications compared with Version 5.2.

Installation

- S7-GRAPH is released for MS Windows 2000 Professional and MS Windows XP Professional.
- As of S7-GRAPH V5.3 there is a new licensing procedure. User rights are no longer issued by means of authorizations but now by means of license keys. License Keys are managed in the Automation License Manager (see User Rights through the Automation License Manager). The "AuthorsW" program is no longer used.

2 Installation

2.1 Automation License Manager

2.1.1 User Rights Through The Automation License Manager

Automation License Manager

To use the programming software, you require a product-specific license key (user rights). Starting with STEP 7 V5.3, this key is installed with the Automation License Manager.

The Automation License Manager is a software product from Siemens AG. It is used to manage the license keys (license modules) for all systems.

The Automation License Manager is located in the following places:

- On the installation device for STEP 7.
- As a download from the Internet page of A&D Customer Support at Siemens AG

The Automation License Manager has its own integrated online help. To obtain help after the license manager is installed, press F1 or select the **Help > Help on License Manager**. This online help contains detailed information on the functionality and operation of the Automation License Manager.

Licenses

Licenses are required to use STEP 7 program packages whose legal use is protected by licenses. A license gives the user a legal right to use the product. Evidence of this right is provided by the following:

- The CoL (**C**ertificate **o**f **L**icense), and
- The license key

Certificate of License (CoL)

The "Certificate of License" that is included with a product is the legal evidence that a right to use this product exists. This product may only be used by the owner of the Certificate of License (CoL) or by those persons authorized to do so by the owner.

License Keys

The license key is the technical representation (an electronic "license stamp") of a license to use software.

SIEMENS AG issues a license key for all of its software that is protected by a license. When the computer has been started, such software can only be used in accordance with the applicable license and terms of use after the presence of a valid license key has been verified.

Notes

- You can use the software without a license key to familiarize yourself with the user interface and functions.
- However, a license is required and necessary for full, unrestricted use of the software in accordance with the license agreement
- If you have **not** installed the license key, you will be prompted to do so at regular intervals.

License Keys can be stored and transferred among various types of storage devices as follows:

- On license key diskettes
- On the local hard disk
- On network hard disk

For further information on obtaining and using license keys, please refer to the online help for the Automation License Manager.

Types of Licenses

The following different types of application-oriented user licenses are available for software products from Siemens AG. The actual behavior of the software is determined by which type license key is installed for it. The type of use can be found on the accompanying Certificate of License.

License Type	Description
Single License	The software can be used on any single computer desired for an unlimited amount of time.
Floating License	The software can be used on a computer network ("remote use") for an unlimited amount of time.
Trial License	The software can be used subject to the following restrictions: • A period of validity of up to a maximum of 14 days, • A total number of operating days after the day of first use, • A use for tests and validation (exemption from liability).
Upgrade License	Certain requirements in the existing system may apply with regard to software upgrades: • An upgrade license may be used to convert an "old version X" of the software to a newer version X+. • An upgrade may be necessary due to an increase in the volume of data being handled in the given system.

2.1.2 Installing the Automation License Manager

The Automation License Manager is installed by means of a setup process. The installation software for the Automation License Manager is included on the STEP 7 product CD.

You can install the Automation License Manager at the same time you install S7-GRAPH or at a later time.

Notes

- For detailed information on how to install the Automation License Manager, please refer to its current "Readme.wri" file
- The online help for the Automation License Manager contains all the information you need on the function and handling of License Keys.

Subsequent installation of license keys

If you start the S7-GRAPH software and no license keys are available, a warning message indicating this condition will be displayed.

Notes

- You can use the standard software without a license key to familiarize yourself with the user interface and functions.
- However, a license is required and necessary for full, unrestricted use of the software in accordance with the license agreement
- If you have **not** installed the license key, you will be prompted to do so at regular intervals.

You can subsequently install license keys in the following ways:

- Install license keys from diskettes
- Install license keys downloaded from the Internet. In this case, the license keys must be ordered first.
- Use floating license keys available in a network

For detailed information on installing license keys, refer to the online help for the Automation License Manager. To access this help, press F1 or select the **Help > Help on License Manager** menu command.

Notes

- In Windows 2000/XP, license keys authorization will only be operational if they are if it is installed on a local hard disk and have write-access status.
- Floating licenses can also be used within a network ("remote" use).

2.1.3 Guidelines for Handling License Keys

Caution

Please note the information on handling license keys that is available in the online help and the Readme.wri on the Automation License Manager. If you do not follow these guidelines, the license keys may be irretrievably lost.

To access online help for the Automation License Manager, press F1 for context-sensitive help or select the **Help > Help on License Manager** menu command.

This help section contains all the information you need on the function and handling of license keys.

2.2 Installation

2.2.1 Installing S7-GRAPH

S7-GRAPH contains a Setup program that installs the package automatically. Prompts on the screen guide you step by step through the entire installation. The Setup program is started as usual when installing software in Windows.

Requirements for Installation

- Microsoft Windows 2000/XP as the operating system
- SIMATIC STEP 7 standard package (for the required version, refer to the Readme.wri file).
- PC or programming device equipped as explained in the readme file

A programming device (PG) is a personal computer designed specifically for use in an industrial environment. It is fully equipped for programming SIMATIC programmable controllers.

- Disk space:
  For the space required on your hard disk, refer to the "Readme".
- MPI interface (optional):
  You only require the MPI interface between the programming device or PC) and PLC if you want to communicate with the PLC via MPI. For this connection you require either:
  - a PC/MPI cable connected to the communications port of your device or
  - an MPI board installed in your device.

  An MPI port is integrated in some programming devices.
- External Prommer (optional):
  An external prommer is only required when you want to blow EPROMs using your PC.

2.2.2 Starting the Installation Program

Preparations for Installation

Before you start the installation, Windows must first be started.

To install from CD-ROM, insert the CD-ROM in the CD-ROM drive of your PC.

Starting the Installation Program

To install the software:

1. In Windows, open the dialog for installing/uninstalling programs by double-clicking the "Add/Remove Programs" icon in the "Control Panel".
2. Follow the instructions for installing the program step by step.

The program guides you step by step through the installation. You can move on to the next step or back to the previous step.

During installation, you will be prompted to answer questions and to select options. Please read the following notes so that you have the information you require for these dialog boxes.

If you already have a version of S7-GRAPH installed ...

If the installation program detects an S7-GRAPH installation on your programming device/PC, a message is displayed and you then have the following options:

- Cancel the installation (you can then uninstall the old S7-GRAPH version under Windows and then start the new installation again) or
- Continue the installation and overwrite the old version with the new version.

In the interests of "clean" data management, you should uninstall an older installation before installing the new version. Simply overwriting an older version also has the disadvantage that when you later uninstall the program, existing parts of the older installation will not be removed.

2.2.3 Notes on Installation

Installing License Keys

During setup, the program checks to see whether a corresponding license key is installed on the hard disk. If no valid license key is found, a message stating that the software can be used only with a license key is displayed. If you want, you can install the license key immediately or continue setup and then install the key later. If you want to install the license key now, insert the authorization diskette when prompted to do so.

Errors During Installation

The following errors lead to the installation being canceled:

- If an initialization error occurs immediately after starting Setup, the most probable reason is that *Setup* was not started under Windows.
- Not enough space on the disk: You must have enough space on your hard disk for the type of installation you have selected (see readme).
- Defective CD/diskette: If you discover that a CD/diskette is defective, please contact your Siemens distributor or sales office.
- If you make a mistake: Start the installation again and follow the instructions carefully.

Completion of the installation ...

Successful installation is indicated by a message on the screen.

If changes were made to DOS files during installation, you will be prompted to restart Windows. Following the restart, you can also start S7-GRAPH.

Otherwise, you can start S7-GRAPH immediately after the setup.

2.2.4 Uninstalling S7-GRAPH

To uninstall the software, use the usual method under Windows:

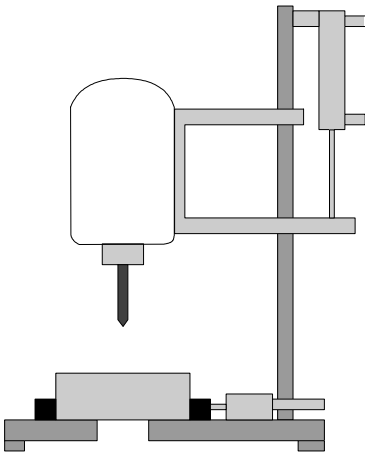
1. Under Windows, open the dialog for installing/uninstalling programs by double-clicking the "Add/Remove Programs" icon in the "Control Panel".
2. Select the S7-GRAPH entry in the list of installed programs. Click the "Add/Remove" button to uninstall the program.

If the "Remove Released File" dialogs appear, click the "No" button if you are unsure whether you want to remove the file or not.

3 Designing a Sequential Control System Based on the Example of a Drill

Welcome to the S7-GRAPH Beginner's Example

If you take an hour to work through the beginner's example, you will learn how to create a sequential control system for automating the drill as described below.



You first learn how to configure a sequential control system efficiently and will then be guided step by step through the tasks you need to perform in the SIMATIC Manager and in S7-GRAPH so that you can

- create the sequential control system,
- download it to the CPU and
- test it.

The correctly programmed sample is supplied with S7-GRAPH as a project called "ZEn02_01_S7GRAPH_Drill".

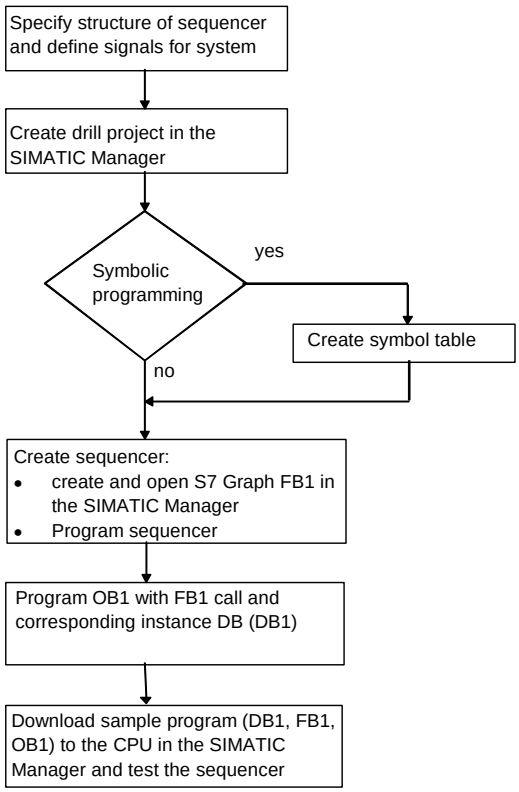
Requirements

To allow you to program and test the "drill" example, you require the following hardware and software:

- Programming device/PC with
  - STEP 7 standard package and the S7-GRAPH optional package
  - MPI connection to the programmable logic controller
- A programmable logic controller (in our example an S7-300) consisting of the following: standard rail, 24V power supply, CPU 314, and a digital input/output module (8DI + 8DO)
- As an alternative to the PLC: The "PLC Simulation" S7 optional package

Procedure for Creating a Sequential Control System

The flow diagram illustrates the procedure for creating the sequential control system for the drill example:



3.1 Technological Task and Functional Diagram

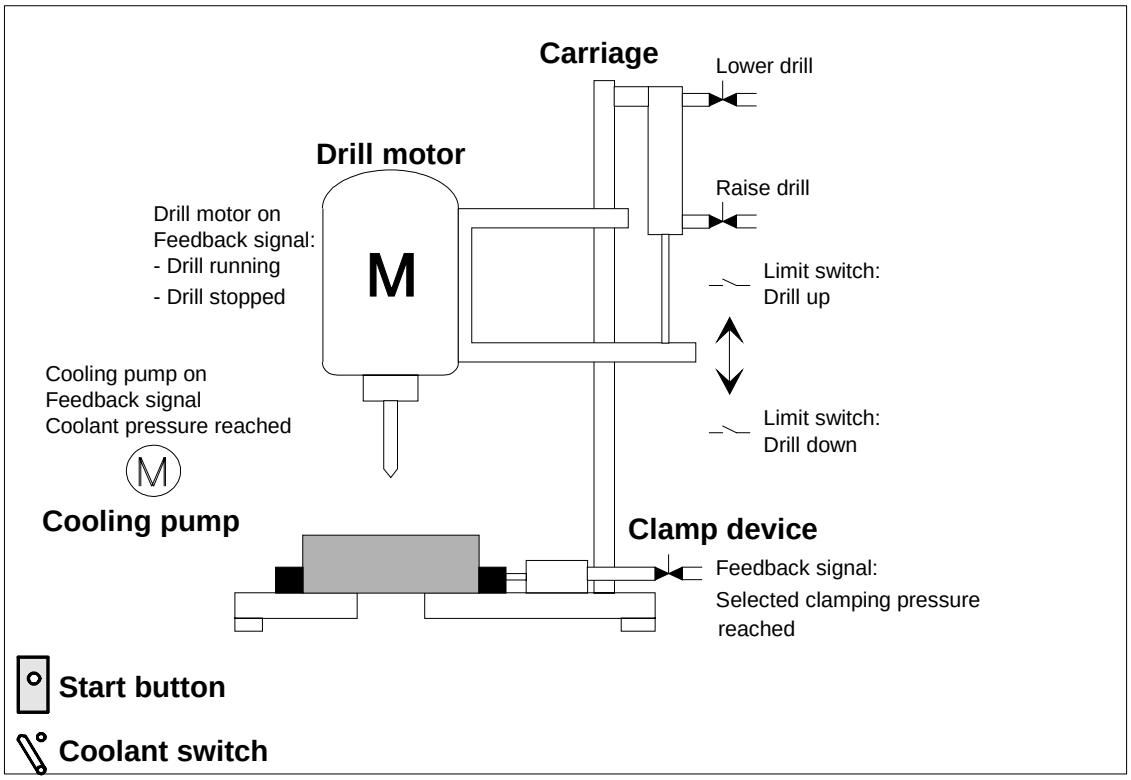
Task

You want to program a sequential control system to automate a drill. The setup of the drill is shown by a technological drawing and the process sequence in the form of a function diagram.

Technological Drawing - Set Up of the Drill

The drill consists of the following elements:

- Drill motor with feedback signals for drill running/stopped
- Start button and coolant switch
- Cooling pump with feedback signal for coolant pressure reached
- Clamping device with feedback signal for selected clamp pressure reached
- Carriage raise/lower drill with limit switches for drill up/down



Initial State

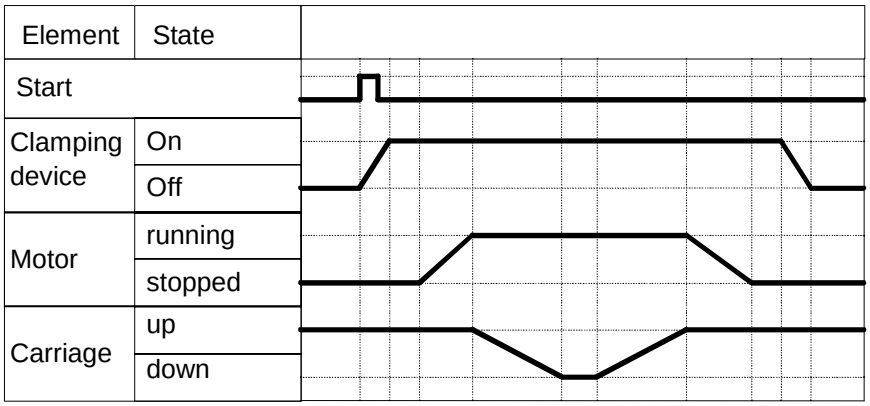
The initial state of the drill is defined as follows:

- The drill motor and cooling pump are stopped
- The carriage/drill is in the upper position
- There is no work piece in the clamping device

Functional Diagram - Drilling Sequence

The entire drilling sequence can be divided into the following sections:

- Insert work piece (manual)
- If required, set switch for coolant (depending on the material)
- Start the machine with the start button (drill motor starts up)
- Clamp the work piece with the selected clamp pressure
- Start the cooling pump (if coolant selected)
- Lower drill and carriage to the bottom target position (drill)
- Wait 0.5 seconds at lower target position (drill)
- Raise drill with carriage to upper target position
- Remove work piece, turn off drill motor and cooling pump
- Remove work piece (manual)



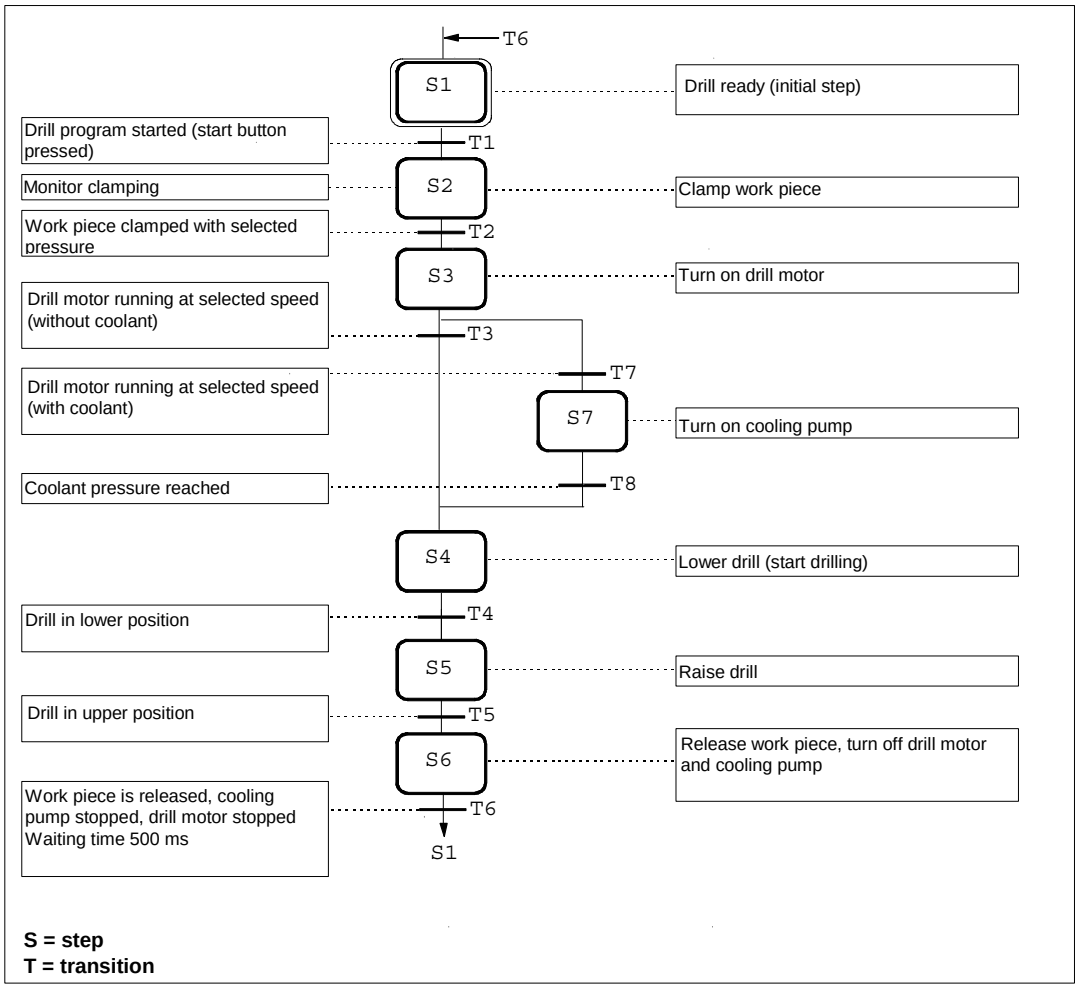
3.2 Selecting the Structure of the Sequencer

Before you create the program for the sequencer, you should include a concept phase in which you break down the drilling operation into single steps. The basis of the concept design is the technological drawing and the flowchart.

Dividing the Drilling Process into Individual Steps - Structure of the Sequencer

The drilling process is described by S7-GRAPH in the form of a sequencer. A sequencer represents a sequence of single steps and conditions that control how the process moves on to the next single step. To specify the structure of the sequencer, follow the steps outlined below:

1. Break down the drilling process into steps and specify the order of the steps (for example "step S2 follows S1" or "step S3 follows either step S4 or S7").
2. For each step, specify the actions that must be performed in the step (for example in S1 the action "Drill ready" or in S3 the action "Turn on drill motor").
3. Then decide for every step which conditions must be met so that the process can move on to the next step (for example for T1 the condition "Drill started - start button pressed" or for T5 the condition "Drill in upper position").



3.3 Defining the System Signals

Once you have broken down the drilling operation into individual steps, you should then define the input and output parameters for each step. The basis of the concept design is the technological drawing and the flowchart.

Defining Inputs and Outputs

List the inputs and outputs for the drill in the form of a table.

If you want to program symbolically, enter the required symbolic names (for example input I 0.4 "Cl_press_ok") for the absolute inputs and outputs and any comments that will make the program easier to understand (for example "Workpiece clamping pressure reached").

In the drill example, it is assumed that the switches and contactors of the drill are controlled via the inputs and outputs of the digital input/output module of the S7-300 programmable controller. The input/output module has 8 inputs and 8 outputs. The default values of the input and output addresses of the module inserted in slot 4 are as follows: I 0 to I 0.7 and Q =0.0 to Q 0.7.

Absolute Addresses	**Symbolic addresses**	**Explanation**
Inputs	**in the program (I)**	
I 0.0	Dr_mot_running	Feedback signal for drill running at selected speed
I 0.1	Dr_mot_stopped	Feedback signal for drill stopped
I 0.2	Drill_down	Limit switch for drill in lower position
I 0.3	Drill_up	Limit switch for drill in upper position
I 0.4	Cl_press_ok	Feedback signal for work piece clamping pressure reached
I 0.5	Coolant_sel	Selector for coolant (dependent on work piece)
I 0.6	Cool_press_ok	Feedback signal for coolant pressure reached
I 0.7	Start_button	Start button of the drill
Outputs	**in the program (Q)**	
Q 0.0	Dr_mot_on	Turn on drill motor
Q 0.1	Cool_pump_on	Turn on cooling pump (dependent on work piece)
Q 0.2	Lower_drill	Lower drill and carriage to bottom limit position
Q 0.3	Raise_drill	Raise drill and carriage to the upper limit position
Q 0.4	Clamp_workp	Clamp work piece at required pressure

3.4 Creating the Drill Project in the SIMATIC Manager

Creating a Project

Projects for sequential control systems do not differ from other projects in STEP 7.

To create a new project in the SIMATIC Manager, follow the steps outlined below:

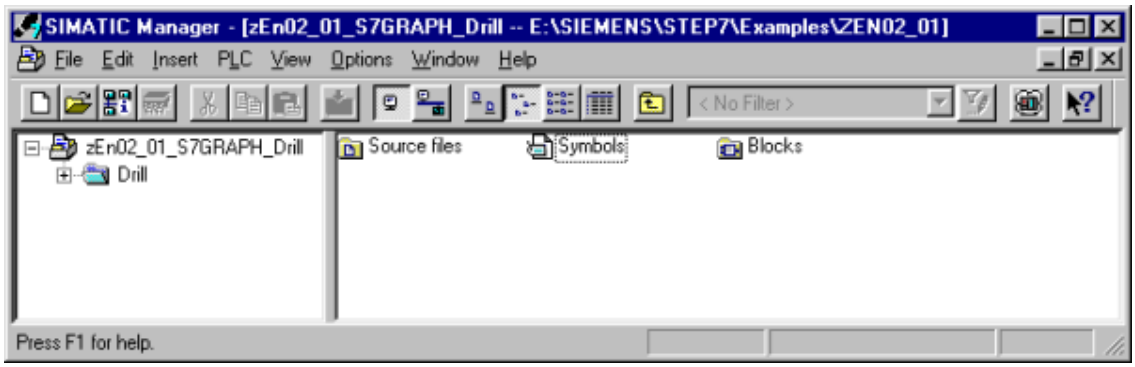
1. Select the menu command **File > New**.
2. Name the project "Drill".

Inserting an S7 Program

In this example, hardware configuration is unnecessary since the default addressing of the input/output module in slot 4 is used. You can therefore immediately insert an S7 program in the project folder in the SIMATIC Manager. The S7 program serves as a folder for the blocks of the user program, the source files and the symbols. Follow the steps below:

1. Select the "Drill" project.
2. Select the menu command Insert > Program > S7 Program.
3. Name the S7 program "Drill Program".

The folders for source files, blocks, and symbols are created automatically when you insert an S7 program. An empty OB1 is also created in the Blocks folder.



3.5 Creating a Symbol Table

When you program in STEP 7, you work with addresses such as I/O signals, memory bits, counters, timers, data blocks, and function blocks. You can access these addresses in your program in absolute format (for example I1.1, M2.0, FB21).

You will find the program much clearer and easier to read if you use symbols (for example Motor_A_on) instead of the absolute addresses. To allow the use of symbols, you can enter a name, the absolute address, the data type and a comment for every address used.

Once you have defined a symbol, it can be used throughout the entire user program of a programmable module.

Creating a Symbol Table

If you want to write your program using symbolic addresses, it is advisable to create the symbol table at this point.

1. Open the symbol table in the "Drill Program" folder by double-clicking "Symbols".
2. Edit the table as shown below.
3. Save the symbol table using the menu command **Table > Save**.

The entries 1 to 14 are required for symbolic representation of the inputs and outputs. The entries 15 to 18 allow symbolic representation of the blocks.

Symbol Editor - ZEn02_01_S7GRAPH_Drill\Drill\Symbols

Symbol Table Edit Insert View Options Window Help

ZEn02_01_S7GRAPH_Drill\Drill\Symbols

	Symbol	Address	Data type	Comment
1	Dr_mot_running	I 0.0	BOOL	Drill motor running at selected speed
2	Dr_mot_stopped	I 0.1	BOOL	Drill motor stopped
3	Drill_down	I 0.2	BOOL	Limit switch "drill in lower position"
4	Drill_up	I 0.3	BOOL	Limit switch "drill in upper position"
5	Cl_press_ok	I 0.4	BOOL	Work piece clamping pressure reached
6	Coolant_sel	I 0.5	BOOL	Cooling pump selector switch on
7	Cool_press_ok	I 0.6	BOOL	Cooling pump running
8	Start_button	I 0.7	BOOL	Start button of the drill
9	INIT_SQ	M 0.0	BOOL	Sequencer parameter: INIT_SQ
10	Dr_mot_on	Q 0.0	BOOL	Turn on drill
11	Cool_pump_on	Q 0.1	BOOL	Supply coolant
12	Lower_drill	Q 0.2	BOOL	Lower drill
13	Raise_drill	Q 0.3	BOOL	Raise drill
14	Clamp_workp	Q 0.4	BOOL	Clamp work piece at required pressure
15	IDB_Seq_drill	DB 1	FB 1	Instance DB of the drill sequencer
16	Seq_drill	FB 1	FB 1	Drill sequencer
17	Cyclic_prg	OB 1	OB 1	Cyclic program
18	TIME_TCK	SFC 64	SFC 64	Time Tick
19				

Press F1 to get Help.

3.6 Creating an S7-GRAPH FB and Programming a Sequencer

Creating an S7-GRAPH FB

The S7-GRAPH FB will contain the sequencer. To create an S7-GRAPH FB, follow the steps outlined below:

1. Open the "Blocks" folder in "Drill Program" in the SIMATIC Manager.
2. Select the menu command Insert > S7 Block > Function Block.
3. Set "S7-GRAPH" as the language in the "Properties" dialog box.

   Result: The empty FB with the default number 1 is created in the "Blocks" folder.

Programming a Sequencer

After you have started the S7-GRAPH editor by double-clicking FB1 the system inserts the first step (initial step) and the first transition. It is advisable to create the structure at the "Sequencer" display level. To display conditions and actions, activate the **View > Display With > Conditions and Actions** menu command.

Using the mouse and the "Sequencer" toolbar at the left-hand edge of the screen, you can then position all the remaining **steps** and **transitions**, the **alternative branch** and the **jump** from the end of the sequencer to the start of the sequencer. There are two ways of doing this and both are illustrated here.

Method 1: "Direct" Mode

1. Select transition 1 and click with the mouse until the icon

    **insert step + transition**

   until you arrive at step/transition 6.
2. Select step 3 and then select the icon

    **open alternative branch**.

   This opens the alternative branch for supplying coolant. The branch begins with transition 7.
3. With the mouse, select the icon

    **insert step + transition**,

   and insert step 7 (S7) and transition 8 (T8).
4. Select the icon

    **close alternative branch**

   and then select transition 3.
5. Now complete the sequencer structure by first selecting transition 6 and then clicking on the icon

    **insert jump**

   and then selecting step 1.

Method 2: "Drag-and-Drop"

1. Return to the SIMATIC Manager and create function block FB2 in the "Blocks" folder as described above. Once again select "GRAPH" as the source language.
2. Start the S7-GRAPH editor by double-clicking FB2 in the "Blocks" folder.
3. Select the menu command **Insert > Drag-and-Drop**.
4. With the mouse, select the icon
    **insert step + transition**
   and then click on the last transition of the individual elements until you arrive at step/transition 6.
5. Select the icon
    **open alternative branch**
   and open the alternative branch for the coolant by clicking on step 3. The branch begins with transition 7.
6. With the mouse, select the icon
    **insert step + transition**,
   to insert step 7 (S7) and transition 8 (T8).
7. Select the icon
    **close alternative branch**
   and first select transition 8 and then transition 3.
8. Now complete the sequencer structure with
    **insert jump**
   by first selecting transition 6 and then step 1.

Note

Close fFB2 before you start to program step actions.. You only created this FB to try out the second method of creating a sequencer structure. When you close the FB, answer all prompts with "No". For the rest of the exercise, you will be working in FB1.

3.7 Programming Step Actions

There are also two methods available for programming step actions and transitions: **Direct** and **Drag-and-Drop**. The procedure described below assumes you have selected the menu command **Insert > Drag-and-Drop**:

1. Select the menu command **Insert > Action**.
   Result: The mouse pointer then appears as shown below:
   
2. Insert an empty action line by clicking the action box.
3. Enter the actions.

An action consists of an instruction and an address. For the drill program, four different instructions are necessary in the steps:

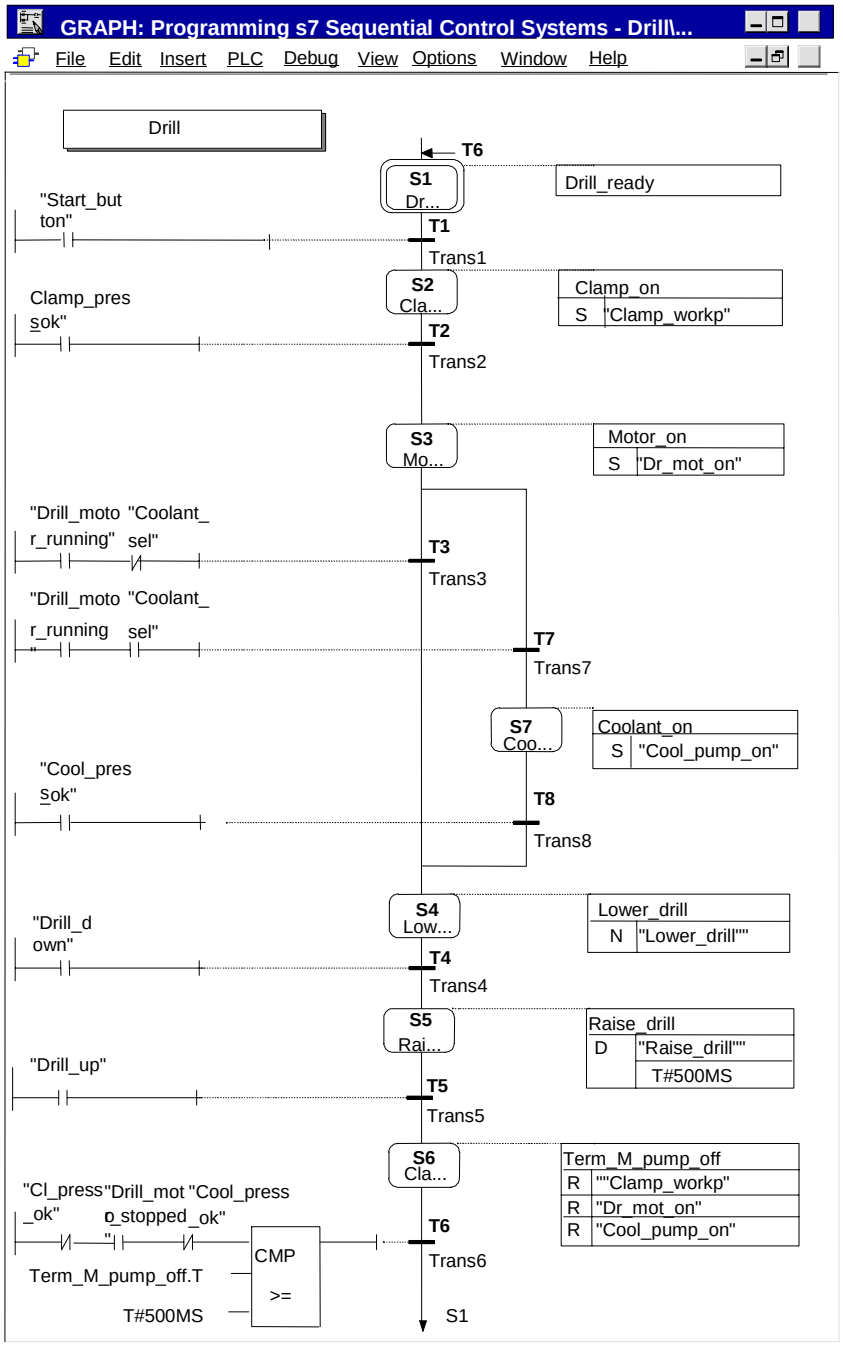
- S Set output
- R Reset output
- N Non holding: As long as the step is active, the signal state of the address is 1.
- D Delay: The address is set to 1 after the defined time has elapsed following activation of the step and is reset when the step is deactivated.

3.8 Programming Transitions

The bit logic instructions "normally open contact", "normally closed contact" and "comparator" are used for the step enabling conditions in the transitions. To program transitions:

1. Set the "LAD" view and select the appropriate icons in the "LAD/FBD" toolbar
    insert normally-open contact
    insert normally-closed contact
    insert comparator
2. Position the symbols at the appropriate points by clicking the transition lines. You can exit the insert mode at any time with the ESC key.
3. Enter the addresses. Click the placeholder "??.?" of the required text field. Then enter an absolute or symbolic address (for example I 0.7, "Start_switch").
4. If you wish, you can also enter a comment for the sequencer. In the "sequencer" view, the comment field is at the top left and can be opened by clicking it with the mouse.

The following figure shows the completed sequencer.



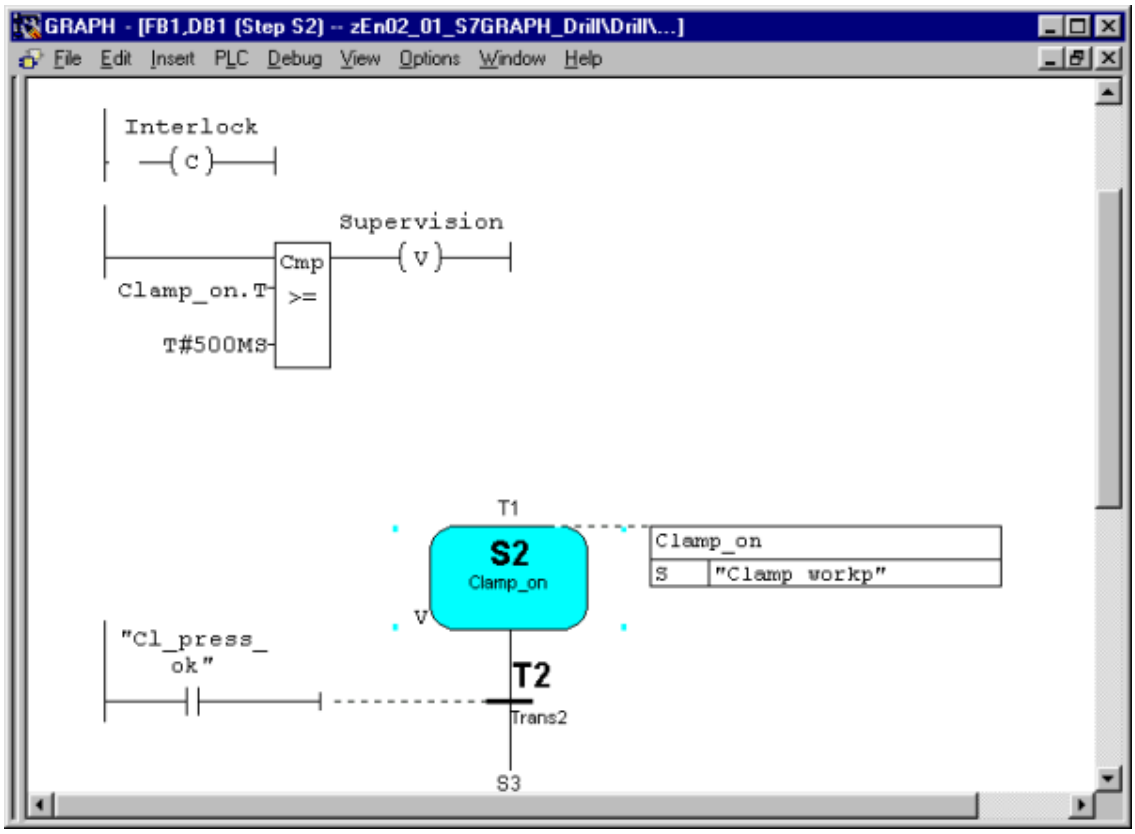
When programming comparators, you can use the system information for steps as the addresses. The addresses have the following significance:

- **Step_name.T**: current or last activation time of the step
- **Step_name.U**: current or last activation time of the step without the time of a disturbance

3.9 Programming Monitoring Functions

To program monitoring functions:

1. Double-click on step 2 to change from the "sequencer" view to the "single step " view.
2. Select the icon
    insert comparator in the "LAD/FBD" toolbar
3. Position the comparator at the appropriate point on the supervision line and enter the required monitoring time.



3.10 Specifying the Standard Function Block

Executability of the S7-GRAPH FB

S7-GRAPH has two options available for creating FBs.

- Full code:
  The entire code required to execute each S7-GRAPH FB is included in the FB. If you have several S7-GRAPH FBs, this means a considerable increase in memory requirements.
- Standard FC required:
  To reduce memory requirements, S7-GRAPH has the two following alternatives: You use a standard FC containing the main code sections for all FBs. This FC is copied to your project automatically when you select this option. The FBs produced using this method are considerably smaller.

For this example, use the compilation option "Full code".

The most suitable FC depends on the performance of your CPU. Select one of the following standard FCs:

FC Number	Functionality
FC72	As default, you work with FC72. Remember, that your CPU must be capable of processing blocks with more than 8 Kbytes.
FC70/FC71	These two FCs are less than 8 Kbytes in size and can therefore be loaded on a smaller CPU. FC70 uses the diagnostic functionality of SFC17/18 and can only be used on CPUs that have these functions available. If your CPU does not have these functions, you must use FC71 and do without diagnostic capability. If you want to check whether or not your CPU contains these SFCs, select the menu command **PLC > Obtainable Nodes** in the SIMATIC Manager or click the corresponding button in the toolbar. Open the "Blocks" folder in the S7 program.
FC73	This block requires less than 8 Kbytes of memory so that it can be executed on all CPUs. Using this FC considerably reduces the memory requirements of the S7-GRAPH FBs. You must also select the option "Interface Description: Memory minimized" in the block settings. The following restrictions, do, however, apply: The blocks created do not have diagnostic capability When you monitor the sequential control system, you will see the status display only for the selected active element.

Setting the Executability

Select the menu command **Options > Block Settings** and specify that the S7-GRAPH FB can be executed with a standard FC in the Compile/Save tab. Enter the FC number that matches the performance of your CPU.

The block is automatically copied to the user project if you select FC70/71, FC72 or FC73 as the block number and there is not yet an FC with this number in the target project. If you want to use a different number for the standard block, you must copy and renumber yourself.

3.11 Saving and Closing the Sequencer

When you save the sequencer, it is automatically compiled.

1. Select the menu command **File > Save**.

Result: The "Select Instance DB" dialog box is opened with the default instance DB (DB1).

2. Accept the settings by clicking "OK".

Result: The instance data block is automatically created in the "Blocks" folder.

Note

The warning "S1 without content" in the compiler log simply means that no action is programmed in step 1.

---

3. To close the sequencer, select the menu command **File > Close**.

3.12 Including the Sequencer in the STEP 7 Program

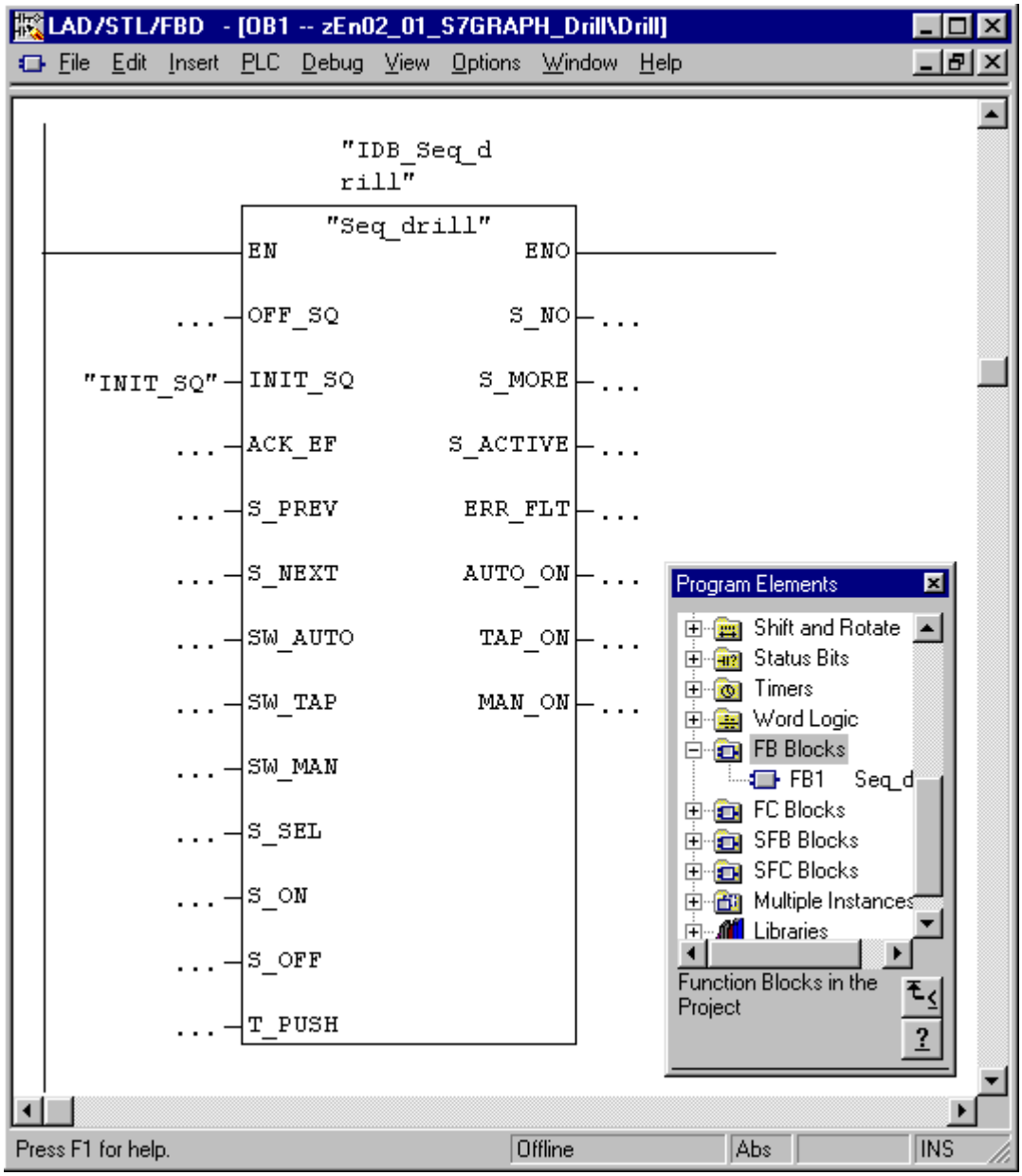
Programming OB1

The sequential control program for the drill is called and started in organization block OB1. You can create OB1 in LAD, FBD, STL or SCL (here it was created in LAD). Program OB1 as shown in the following diagram. Follow the steps below:

1. Open the "Blocks" folder in the "Drill Program" S7 program in the SIMATIC Manager.
2. Start the LAD/STL/FBD editor by double-clicking OB1.
3. With the menu command **View**, select the programming language LAD.
4. Select segment 1 and insert the sequencer call using the program element catalog by double-clicking FB1 (Seq_drill).
5. Type in the name of the corresponding instance data block (IDB_Seq_drill) above the LAD box.
6. Select the input parameter INIT_SQ, insert a normally open element using the "LAD" toolbar and label it M0.0 ("INIT_SQ"). Using this parameter, you can set the sequencer to the initial step (in the example step 1) in the online mode.
7. Select the menu command **File > Save** and close the organization block with the menu command **File > Close**.

Note

All other block parameters can be ignored for the example.


LAD/STL/FBD - [OB1 -- zEn02_01_S7GRAPH_Drill\Drill]
File Edit Insert PLC Debug View Options Window Help
"IDB_Seq_drill"
"Seq_drill"
EN
ENO
OFF_SQ
S_NO
"INIT_SQ"
INIT_SQ
S_MORE
ACK_EF
S_ACTIVE
S_PREV
ERR_FLT
S_NEXT
AUTO_ON
SW_AUTO
TAP_ON
SW_TAP
MAN_ON
SW_MAN
S_SEL
S_ON
S_OFF
T_PUSH
Program Elements
Shift and Rotate
Status Bits
Timers
Word Logic
FB Blocks
FB1 Seq_d
FC Blocks
SFB Blocks
SFC Blocks
Multiple Instances
Libraries
Function Blocks in the Project
Press F1 for help.
Offline
Abs
INS

3.13 Downloading the Program to the CPU and Testing the Sequencer

Downloading the User Program

To allow you to download the program to the CPU, you must download all the blocks (DB1, FB1, OB1, FC70/71, FC72 and/or. FC73) to the CPU of the programmable controller in the SIMATIC Manager. Follow the steps outlined below:

1. Open the "Drill Program" S7 program in the SIMATIC Manager and select the "Blocks" folder.
2. Select the menu command **PLC > Download**.

Caution

It is best to download the S7-GRAPH block in the STOP mode since following downloading of the instance DB, the sequencer is automatically set to the initial state.

You should only download S7-GRAPH blocks in the RUN-P mode when the sequencer is in the initial state or in the OFF state. If you download the blocks the sequencer in a different state, for example when overwriting an old block, problems may occur in the synchronization of the sequencer with the process.

Testing the User Program

To test the user program, you require an online connection to a CPU.

1. Open the project window in the SIMATIC Manager.
2. Open the sequencer by double-clicking FB1.
3. Select the menu command **Debug > Monitor**.

   Result: The program status is displayed (the initial step is active). Active steps are displayed in color.

Caution

A monitoring time is programmed in step 2. If the step activation time exceeds the configured monitoring time (500 ms) in the supervision condition, the system recognizes a supervision error and the disturbed step is displayed in red. If a fault occurs, you must first satisfy the condition for progressing to the next transition. Using the PG function **Debug > Control Sequencer** you can then enter an acknowledgment (see also "Control Sequencer").

This does not apply to the inching mode, since the step enabling condition must be satisfied and the acknowledgment received within one cycle.

3.14 Test Function: Control Sequencer

Control Sequencer is a test function with which you can test the sequencer in S7-GRAPH in all modes. All the settings and entries for the dialog box have the same effect as the corresponding FB parameters.

The entries in the "Control Sequencer" dialog box can be different from the settings you used to compile the sequencers. The dialog box settings have priority.

"Control Sequencer" Dialog box

The "Control Sequencer" dialog box is used both as an output field that displays the current settings and as an input field in which you can change the current status.

If you acknowledge an error, initialize the sequencer, or want to change the step in the manual mode, call the dialog box with the menu command **Debug > Control Sequencer**.

Acknowledge

If the "Acknowledge errors" option is clicked, you acknowledge a pending error message with the "Acknowledge" button. In this way, you can acknowledge a disturbance caused, for example, by the configured monitoring time in Step 2 being exceeded.

Note, however, that before you acknowledge the error you must make sure that the supervision or interlock conditions that led to the error are no longer satisfied.

After an error has occurred, you can enable the next step in the sequencer by satisfying the step enable condition because the last step enable condition in the cycle in which you acknowledge has priority over the supervision. You must nevertheless acknowledge the error.

If the error was caused because the undisturbed activation time **stepname.U** has exceeded the configured monitoring time, you can move on the sequencer by clicking the "Acknowledge" button.

This is possible because the **Step name.U** variable is set to "0" when you acknowledge.

Initialize

With the "Initialize" button, you can restart the sequencer completely with the defined initial step.

Setting the Mode

Checkmark on the check boxes to set the corresponding selected mode for the sequencer

You can, for example, run your sequencer in the manual mode instead of the automatic mode. In the manual mode, you can activate or deactivate any step. Follow the steps outlined below:

1. Enter the step number of the step you want to process in the "Step number" box or simply click the required step.
2. Select the action to be executed with the step:
   - **Activate** The selected step is activated even if the preceding transition is not satisfied.
   - **Deactivate** The selected step is deactivated.

Note that before you activate a step you must first deactivate the currently active step since only one step can be active at any one time in a linear sequencer. In other words, you can only activate the required step when you have deactivated the currently active step.

If you have set a new mode, the original mode is indicated in bold face.

After you have controlled your sequencer in the manual mode, you can return to the automatic mode and close the "Control Sequencer" dialog box.

3.15 Modifying the Sequencer

Often, a program does not run as required at the first attempt. This means that changes and modifications are required. This is also the case in our sample program. The configured watchdog time in Step 2 is too short and the program hangs at this step. The following section explains how to make the required changes. You must first change to the "offline" mode.

Modifying the Watchdog Time

The watchdog time configured in Step 2 is 500 ms. To give you more time for setting the "Cl_press_ok" variable, change the watchdog time to 5 s as explained below:

1. Double-click on step 2 to change from the "sequencer" view to the "single step " view.
2. Close the status by selecting the menu command **Debug > Monitor** or clicking the following button:  .
3. Enter the value 5 s for the monitoring time.
4. Save the sequencer with the menu command **File > Save**, since the sequencer must be recompiled.

Downloading the Modified Sequencer

Once you have compiled the sequencer, you must download it to the CPU. You can download the sequencer in the SIMATIC Manager or in the S7-GRAPH editor. In the S7-GRAPH editor, follow the steps outlined below:

1. With the FB open, select the menu command **PLC > Download** or click on the **Download** button  . The "Download" dialog box is opened.
2. Accept the defaults by clicking "OK".
3. Confirm the prompt to overwrite the existing blocks.

Include Instance DB

If you have not cleared the system default "Include Instance DB", S7-GRAPH checks whether the instance DB has changed since it was last downloaded and whether it must be updated. The option "Download with Instance DB" is enabled in the "Download" dialog box.

Otherwise, S7-GRAPH does not check the instance DB and leaves you to decide whether or not to download the instance DB again. The option "Download with Instance DB" is not clicked in the "Download" dialog box.

Restarting the Status

To be able to monitor or control the sequencer again, restart Status by selecting the menu command **Debug > Monitor** or by selecting the **Monitor**  button.

4 Working with an S7-GRAPH

4.1 Starting S7-GRAPH

Requirements for Running S7-GRAPH

- The STEP 7 standard package is installed on your PC/programming device.
- S7-GRAPH is installed on your PC/programming device.
- You have installed the authorization for S7-GRAPH.

Starting S7-GRAPH

There are two ways of starting S7-GRAPH:

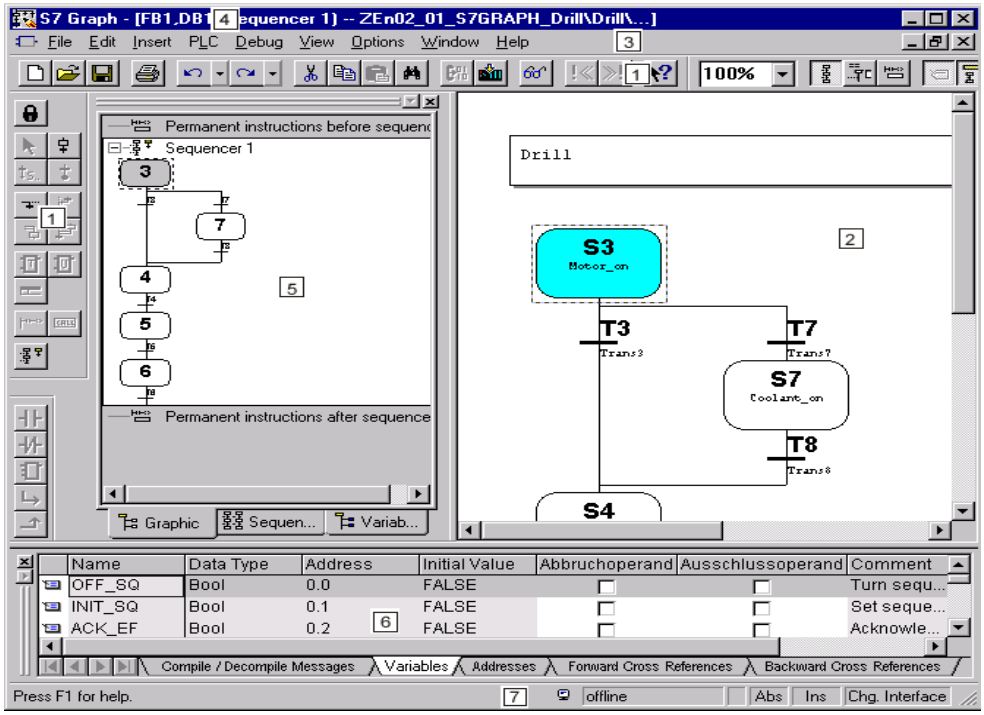
- Startup under Windows:
  Start S7-GRAPH by via the Windows "Start" button on the taskbar.
  S7-GRAPH is found under **Simatic > STEP 7 > S7-GRAPH - Programming Sequential Control Systems**.
- Startup via SIMATIC Manager
  In the SIMATIC Manager, double-click an S7-GRAPH FB or an S7-GRAPH source file.
  S7-GRAPH opens and displays the selected block in the default view.

4.2 The S7-GRAPH User Interface

Elements of the S7-GRAPH User Interface

The user interface is split into three sections:

- The working area (2) shows the sequencer to be edited. Here you can define the structure of sequencers or you can program individual actions and conditions.
- The overview window (5) provides an overview of the global structure of the sequential control, of block parameters and variables, as well as of the environment of the function block in the S7 program.
- The detail window (6) provides specific information you require during specific phases of programming, e.g. compiler messages or information on address instances.



These sections describe the elements of the user interface numbered in the figure.

(1) Toolbars

(2) Work area

(3) Menu bar

(4) Title bar

(5) Overview window

(6) Detail Window

(7) Status bar

Toolbars

The toolbars contain buttons with which you can start frequently required and currently available menu commands.

The names of the toolbars in S7-GRAPH are visible when the toolbars are positioned within the main window.

Name	Content
Standard	Contains functions for handling files (opening, saving etc.) and for editing (copying, pasting etc.).
View	Contains symbols for selecting various views (for example how S7-GRAPH FBs are displayed, display of symbolic names etc.).
Sequencer	Contains buttons for editing the sequencer (for example inserting a step transition pair).
LAD/FBD	Contains buttons for inserting Ladder Logic/FBD elements (for example comparator).

Working Area

The working area is a window within S7-GRAPH in which the sequences of an S7-GRAPH FB or an S7-GRAPH source file are displayed and can be edited.

Within the main window of S7-GRAPH, you can open several at the same time. You can cascade, tile or overlap these windows.

Menu Bar

Just like other Windows applications, S7-GRAPH has a menu bar below the title bar of the main window. The display of the menu commands is context-sensitive. The menu commands are either active (black) or inactive (gray) depending on whether or not the command can be applied to the currently selected object.

Title Bar

The title bar at the upper edge of a window contains the standard Windows buttons and the following information:

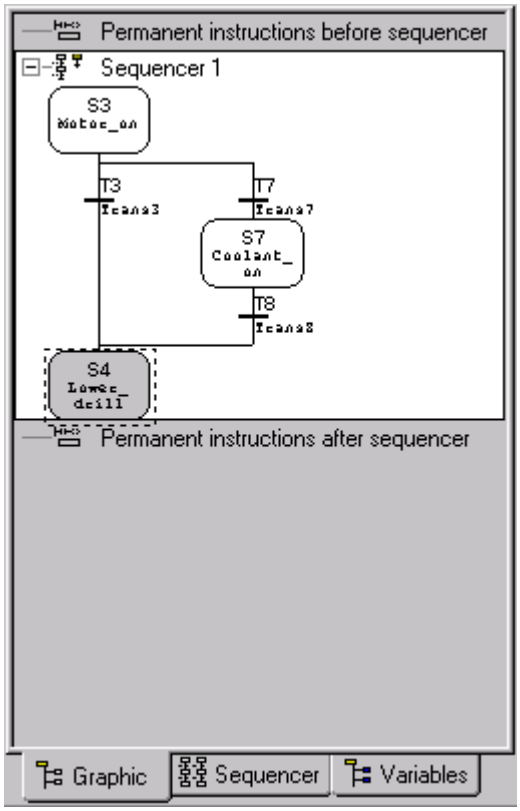
- Title of the software
- Number of the open FB and DB or name and number of the open source
- Displayed element
- Path of the block or source

Overview Window

This window is snapped to the left edge of the working window by default. However, you can also snap it to the right, upper or lower edge of the window.

This window contains three tabs:

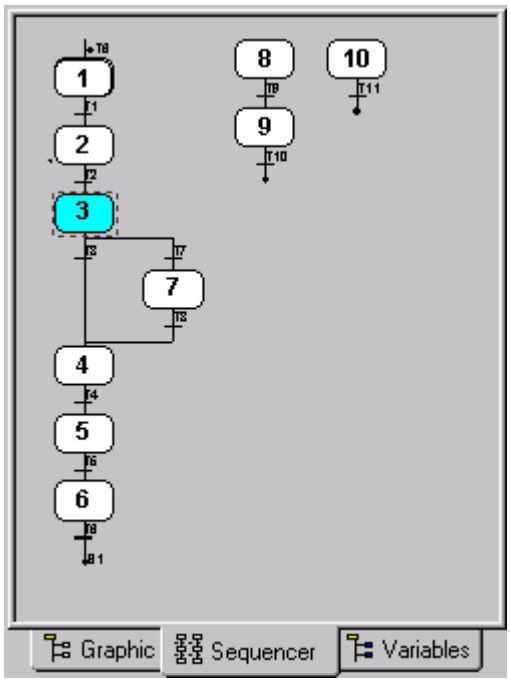
"Graphics" Tab



These tabs display the individual sequencers and permanent operations in textual form. If required, If required, you can expand these to show more graphical detail, e.g. the structure of a sequence or specific operations.

Here you can copy, cut, past or delete complete sequences or elements of such. An element of the sequencer's structure you select from the "Graphics" tab will be displayed for editing in the working window.

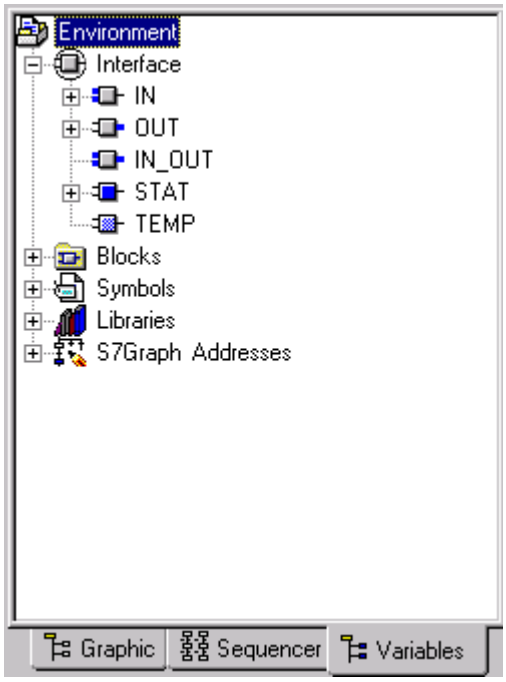
"Sequences" Tab



This tab displays all sequencers in a well arranged tiled view. This kind of display is especially useful for the status view.

Here, too, you can select an element of the sequencer for editing in the working window. In the actual "Sequences" tab you can not edit sequencers.

"Variables" Tab



Displays the variable declaration with the IN, OUT, IN_OUT, STAT and TEMP segments and further program elements: Symbols from the symbol table, preprogrammed blocks from the block folder and blocks from libraries. A further folder displays the steps and transitions - including their components - of the S7-GRAPH function block which you can use as addresses in the program.

In this tab, you can modify existing parameter sets, system parameters can be deleted but cannot be edited. It is also possible to extend your own parameters.

If you want to use a parameter or a program element in the sequencer, you can drag it from the variable declaration window to the sequencer using the mouse and position it at the required location.

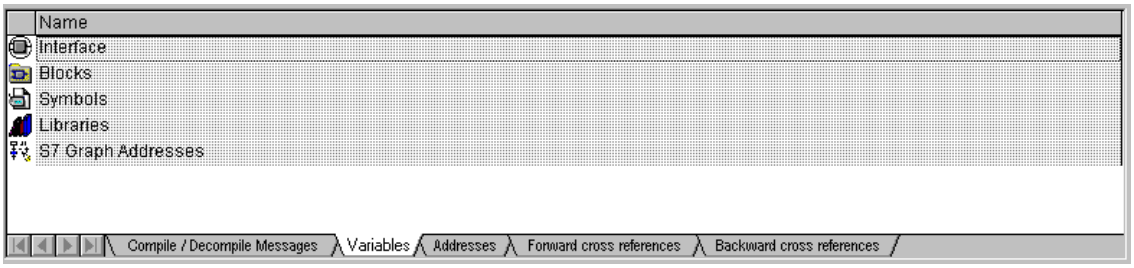
Changes you have made to the parameters in the variable declaration window are automatically updated in the sequencer so that no inconsistencies can result.

With the F1 key, detailed help is available on declaring variables in the variable declaration window.

For more detailed information on the declared variables and program elements, refer to the "Details" window (Variables tab). You can open this window with the menu command **View > Details**.

"Details" Window

The "Details" window is displayed at the bottom edge of the screen and includes the following tabs:



Compile / Decompile Messages

The tab is used to display errors and warnings that occur for example after compiling the sequencer.

Variables

The content of this tab depends on the selection you made in the "Variables" tab of the overview window. Here you can view **detailed information** on:

- Variables
- Symbols from the symbol table
- Programmed blocks from the block folder
- Blocks from libraries
- Steps and transitions of the S7-GRAPH FB

In this tab, you can declare new variables with the menu command **Insert > Declaration Line**.

If you want to use a variable, block parameter or a program element in the sequencer, you can drag it from the variable declaration window to the sequencer using the mouse and position it at the required location.

Addresses

This tab contains a list of the addresses used in the block with their locations, symbols, and data types. During monitoring, the tab displays a further column when "Single Step" or "Permanent Instructions" is set displaying the current status value of the address during program execution.

Whether or not this is displayed depends on the setting for the working area.

Setting for the Working Area	Displayed Addresses
Sequencer	Addresses in the program
Single step display	Addresses of the step and corresponding transition
Permanent operations	Addresses used in the permanent operation

Forward Cross-references

Shows the use of addresses in the memory areas Q, M, T, and C within the user program.

The up references tab displays all the addresses supplied with values in the current block and evaluated at other points in the program.

Backward Cross-references

The down references tab displays all the addresses supplied with the values of other locations in the program and evaluated in the current block.

Status Bar

The status bar is at the bottom edge of the main window of S7-GRAPH. By selecting the menu command **View > Status Bar**, you can display or hide the status bar.

In the left half of the status bar you will see brief information about the selected menu command. The right-hand side of the status bar contains the following information:

The following symbols identify the various connection and operating states:

Symbol	Display	Description	With
	offline	You are editing a block stored on the PG/PC.	S7-300 and S7-400
	DISCONNECTED	The PLC connection is offline.	S7-300 and S7-400
	CONNECTED	You are editing a block that is located on a PLC (SIMATIC S7).	S7-300
	RUN/RUN-P	Module status	S7-400
	STOP	Module status	S7-400
	HALT	Module status	
	FRCE	A force job is busy.	S7-400

The following displays are shown on a colored background in the status bar:

Display	Color
FRCE	Yellow
STOP	Red
HALT	Yellow
Disconnected	White
RUN/RUN-P with "Sequence Control" and "Enable Monitoring"	Green scroll bar with indication: • "AUTO", for automatic mode • "MAN", for manual mode • "TAP", for jog mode • "TOP", for automatic mode or switching to the next step

Group Error

A red bar indicates that an group error has occurred.

Abs/Sym

The address priority set in the properties of the block folder.

The address priority specifies whether the absolute addresses or the symbols are valid if changes are made in the symbol table.

Ins/OVR/Rd

- **Ins**ert/**Ov**e**r**write
  Editing mode for input
- **R**ea**d**
  The block/source is read-only.

Chg/Chg. Interface

Indicates that the block has been changed and whether the change involves the interface.

4.3 Settings for the Work Area

S7-GRAPH offers various display modes and settings that allow you to customize the working area and the user interface, . Some of these settings are saved along with the block you are editing.

S7-GRAPH supports you with different display levels that you can select with the **View** menu:

- Sequencer
- Single-Step Display
- Permanent Instructions

Sequencer

In the "Sequencer" view, one of the sequencers contained in the FB is displayed. If several sequencers exist, you can change from one to the other by selecting the "Graphic" tab in the overview window.

This display level can be used to configure the sequencers. At this level, you can specify the structure of the sequencers and select the numbers, names and comments for steps and transitions.

You can call menu command **View > Display With > Conditions and Actions** to display the contents of the configured steps and transitions.

The following elements are displayed:

- The block comment
- The layout of the steps, transitions, and branches
- As an option, you can display the contents of the steps and transitions
- Depending on the zoom factor, step and transition names are displayed

Single-Step Display

The single-step level always shows only one step-transition pair. All details can be programmed at this display level, as follows:

- Action: Actions executed within the step
- Supervision: Conditions for monitoring the displayed step
- Interlock: Conditions for interlocking the displayed step
- Transition: Conditions for switching from the current step to the next step
- comments
- Numbers and names of the displayed step / transition pair
- Name extensions.
- List of the symbolic addresses

Note

You can use menu command **Edit > Go To** to display a different step without having to move to a different view.

Use the cursor keys to change to the next or previous step.

Permanent Instructions

At the permanent instructions display level, you can program the following:

- **Permanent instructions** before the sequencer
- **Permanent instructions** after the sequencer

Setting a Display Level

To set a display level:

1. Open the **View** menu.
2. Select the display level suitable for the current task:

- **Sequencer** for structuring the sequencer
- **Single step** to program individual steps and transitions.
- **Permanent instructions** for programming permanent conditions and permanent block calls

Default Display Level

An S7-GRAPH FB is always opened at the default display level. You select the default with the menu command **Options > Application Settings** in the "General" tab.

Under "New Window", select the display level at which the S7-GRAPH FB will be opened.

4.4 Customizing a Display Level, Colors and Fonts

Customizing a Display Level

You can adapt the default display level to suit your purposes. S7-GRAPH provides the following options:

- Zoom in and zoom out
- Zoom factor for the current display
- Lasso functions for zooming
- Buttons for handling windows
- Displaying and hiding selected areas
- Setting a default display for certain areas

Zoom In and Zoom Out

Select the menu commands **View > Zoom In** or **Zoom Out** to change the size of the display.

Specifying a Zoom Factor for the Current Display

You can adapt the size of the current display to suit its contents by selecting the menu command **View > Zoom Factor**.

If you select a zoom factor that would make text too small to be legible, the step and transition names are no longer displayed and only the numbers of steps and transitions are visible. You can, nevertheless, display step and transition names by positioning the mouse pointer on the step or transition.

If you want to avoid this, you can specify minimum zoom factors for the various views so that the display size never becomes illegible.

You can also revert to the previous zoom factor at any time.

Lasso Functions for Zooming

To increase the size of a specific area, you can also use the lasso function. Select the menu command **View > Enlarge Active Area**. Then drag a rectangle round the area you want to zoom using the mouse. Now only the selected area is displayed in the S7-GRAPH work area.

Buttons for Handling Windows

S7-GRAPH provides you with a button for displaying and hiding the "Details" and "Overview" windows. This allows you to set your desktop to suit your requirements.

Displaying and Hiding Areas

Apart from displaying individual steps and transitions, S7-GRAPH provides further options in the working area. Select the required menu commands from the **View** menu:

- To display a comment field: **View > Display With > Comments**
- To display addresses in symbolic representation: **View > Display With > Symbols**
- To display conditions and actions: **View > Display With > Conditions and Actions**
- To display a list box for selecting addresses when editing actions: **View > Display With > Symbol List**

Default Display of Additional Elements

Additional elements can also be displayed as part of the default setting when a block is opened. You select the default with the menu command **Options > Application Settings** in the "General" tab. Select the displayed options to make the default setting:

- To display a block comment or a step comment: "Display With: Comments".
- To display addresses in symbolic representation: "Display With: Symbols".
- You can display conditions and actions with **View > Display With > Conditions and Actions**.

Customizing Colors and Fonts

Graph 7 also allows you to customize the appearance of individual elements. These include the settings for color, font, lines, and LAD/FBD addresses.

Modifying the Settings

You can make all the required settings in the "Editor" tab. You can display this tab with the menu command **Options > Application Settings**. When you close the editor (when you quit the customize functions) any modifications you have made are saved.

Color

You can change the color of the following elements:

Element	**Create Mode**	**Monitor Mode**
Selected element	Selected element	
Step active/transition valid		Elements
Step not active		Inactive step
Error	Operator input error	Disturbed step
LAD/FBD satisfied		Satisfied conditions in transition, supervision, interlock and permanent instructions
LAD/FBD not satisfied		Non-satisfied conditions in transitions, supervision, interlock, and permanent instructions
LAD/FBD not processed		Non-executed transitions, supervision conditions or interlocks
Synchronization point		Synchronization points

Font

You can change the font for the following elements:

Area	
Step/transition number	Numbers of the steps/transitions
Step/transition name	Name of the steps/transitions
LAD/FBD	LAD/FBD addresses
The extended name.	Titles of the action tables
Actions	Contents of the action tables
Comments	Contents of the comment fields

Lines

You can change the line style (thickness) for the following types of line:

Line Style	
Normal line	The entry applies to all lines of the sequencers
Sensitive line	Line to which the currently selected Graph element can be appended.

LAD Addresses and Action Lines

You can decide the maximum number of characters permitted for an LAD/FBD address or an action line. You can limit the address to one line in the program view.

Tip: If the LAD/FBD address or the action line is not displayed completely in a particular view, you can display the complete name as a tool tip by positioning the mouse pointer on the relevant address or action.

You can also specify whether the LAD/FBD address is right-aligned and the actions left-aligned.

4.5 Tools for Editing the Sequencer

4.5.1 Display and Position of the Toolbars

Displaying and Hiding Toolbars

Select the menu command **View > Toolbars**. In the dialog box that is displayed, you can select the toolbars you want to be displayed.

Positioning the Toolbars

Toolbars can be positioned anywhere within the main window of S7-GRAPH. You can drag and drop them as required.

1. Click the gray area of the toolbar on which the buttons are arranged and hold down the mouse button.
2. Drag the toolbar with the mouse to the required position on the screen.

Display of the Buttons in Toolbars

The display of the buttons in the toolbars is context-sensitive. That is:

- Enabled buttons are displayed in black color.
- Disabled buttons are grayed out.

4.5.2 Editing the Sequencer with the Toolbars

The way in which you edit a sequencer with the buttons of the toolbar is decided via the menu commands **Insert > Drag-and-Drop** or **Insert > Direct**.

- In the Drag-and-Drop editing mode, you drag an icon from the toolbar to the required position.
- In the Direct mode, you position an icon at the selected point.

"Drag and Drop" editing mode

When you are working in the "Drag and Drop" editing mode, you add icons as follows:

1. Click on a toolbar icon or select an element from the **Insert** menu.

   Result: The mouse pointer takes on the shape of the corresponding icon.

2. Now click the position at which you want to insert the element. If the mouse pointer takes the shape of a "prohibited" sign (circle with a diagonal bar), this indicates that the element cannot be inserted at the selected point. Each click inserts a further element.
3. After you have inserted all the required elements, click on the toolbar icon again or select the menu command from the **Insert** menu again to complete the action.

"Direct" Editing Mode

If you are in the "Direct" editing mode, you insert icons as follows:

1. Select the element after which you want to insert a new element.
2. Click on a toolbar button or select an element from the **Insert** menu.
3. To add further elements of the same type at the same position, click on the button again or select the menu command from the **Insert** menu again.

4.5.3 Navigating in the Sequencer

To allow you to navigate in the sequencer, you can use not only the keys of the cursor pad but also the functions in the **Edit > Go To** menu. These functions make it easier to navigate through large sequencers and when debugging.

Displaying a Specific Step or Transition

With the menu command **Edit > Go To > Step / Transition**, you can jump to a specific step-transition pair. You can specify either the number or the name.

Search for Instances of an Address

You have several options of displaying the program instances of a selected address:

- The **Go To > Instance** function displays a table showing all instances of the selected address in the entire S7 program. The table contains information about the following:
  - Block: Indicates the block in which the address is used
  - Type: Indicates read (R) and/or write (W) access.
  - Language/details: Information in the specific language
- After you double-click on a global address, the S7 program performs a search for "opposing" address instances, that is, for write access addresses it tries to find instances of read access addresses and vice versa.
- The menu commands **Go To > Previous local instance / Next local instance** shows all instances of local addresses in the S7-GRAPH block.

Displaying the Next or Previous Error Location

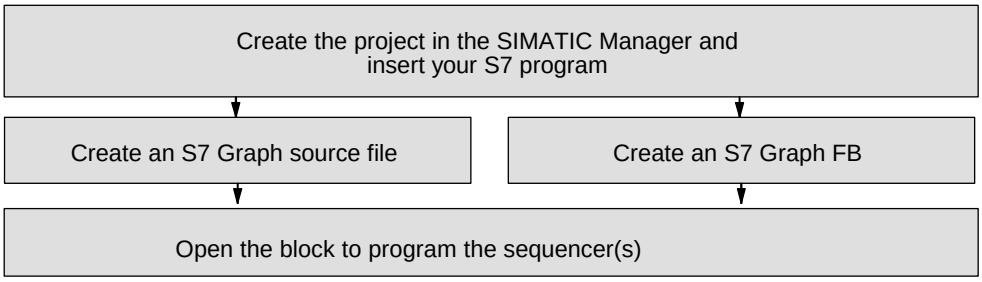
If the message window displays a compilation error, call menu commands **Edit > Go To > Next Error** and **Edit > Go To > Previous Error** to jump to the error location. S7-GRAPH then jumps to the location of the error in the sequencer and displays the error message in the status bar.

5 Creating a Project

This chapter explains how to create and open your project. The chapter also explains decision criteria for working with S7-GRAPH FBs or S7-GRAPH source files.

Steps in Creating a Project

All the information about a sequential control is gathered in a STEP 7 project.



5.1 Creating the Project in the SIMATIC Manager

Before you can create your sequencers with S7-GRAPH, you must first create a project and then an S7 program for the same project. All blocks for the sequential control are saved to this folder.

Creating a New Project

Select menu command **File > New** in SIMATIC Manager to generate a new project file. As of V4x, STEP 7 provides an alternative in the form of a wizard that helps you to create the entire project structure. Select the method you prefer to create a project.

Implementation of an S7 Program in the Project

In SIMATIC Manager, create an S7 program via the menu command **Insert > Program > S7 Program**. The system automatically generates the folder hierarchy for the source files, blocks and for the symbol table. An empty OB1 is generated in the block folder.

5.2 S7-GRAPH FB or S7-GRAPH Source File?

Before you start to create sequencers, you must first choose the type of block in which you want to program. You can create sequencers either in an S7-GRAPH FB or in an S7-GRAPH source file.

Deciding Between an S7-GRAPH FB and an S7-GRAPH Source File

When you save an **FB**, S7-GRAPH performs and implicit compilation of the block, that is, a syntax check on the sequencer you have created. Only error-free S7-GRAPH FBs can be saved and downloaded to the CPU. It is not possible to close an FB that contains errors.

If you are unable to debug the program immediately, you can also save the S7-GRAPH FB as a **source file** or you can first create an S7-GRAPH FB source file instead of an S7-GRAPH FB. In contrast to the S7-GRAPH FB, you can save an S7-GRAPH source file containing errors at any time.

Note

If you find that you cannot program a sequencer in one session, first save the data as an S7-GRAPH source file or create an S7-GRAPH source file in the first place.

5.3 S7-GRAPH FB

You can create an S7-GRAPH FB both in SIMATIC Manager or directly in S7-GRAPH, as follows:

- In S7-GRAPH, select menu command **File > New** to generate a new function block in the S7-GRAPH language.
- In the SIMATIC Manager, select menu command **Insert > S7 Block > Function Block** to generate a new function block. Here you also select the S7-GRAPH language.

Rules for Creating the S7-GRAPH FB

The S7-GRAPH FB is saved to the block folder. The following rules apply:

- You can program the sequencer with absolute and symbolic addresses.
- The FB is displayed depending on its storage locations:
  - offline: The FB in the database on your PG is displayed, including the symbols and comments.
  - online: The FB from the CPU is displayed, including the comments and symbols from the PG database (assuming that these exist offline).
  - online without existing project structure: The FB from the CPU is displayed (view "Accessible Nodes" in the "Open" dialog box) like a STEP 7 FB, in other words without symbols and comments.

5.4 S7-GRAPH Source File

You can create source files both in the SIMATIC Manager, S7-GRAPH or other editors and then import the to an S7 project. Note the description of the syntax.

Rules for Creating and Calling the S7-GRAPH Source File

The S7-GRAPH source file is saved to the "Sources" folder. The following rules apply:

1. Empty S7-GRAPH source files can also be created in the SIMATIC Manager or imported into the open project.
2. An S7-GRAPH source file can also be generated from an open S7-GRAPH FB.
3. Existing S7-GRAPH source files can be opened both in the SIMATIC Manager and in the S7-GRAPH editor. They are then displayed in the S7-GRAPH editor and can be edited.
4. Source files of a project can only be edited offline.

5.5 Opening an S7-GRAPH FB or an S7-GRAPH Source File

You can open an S7-GRAPH FB or an S7-GRAPH source file either in the SIMATIC Manager or in S7-GRAPH.

Note

A new FB created in S7-GRAPH will be opened directly.

Opening in the SIMATIC Manager

In the SIMATIC Manager, proceed as follows:

- Select the required object in the block or source files folder and double-click to open it.

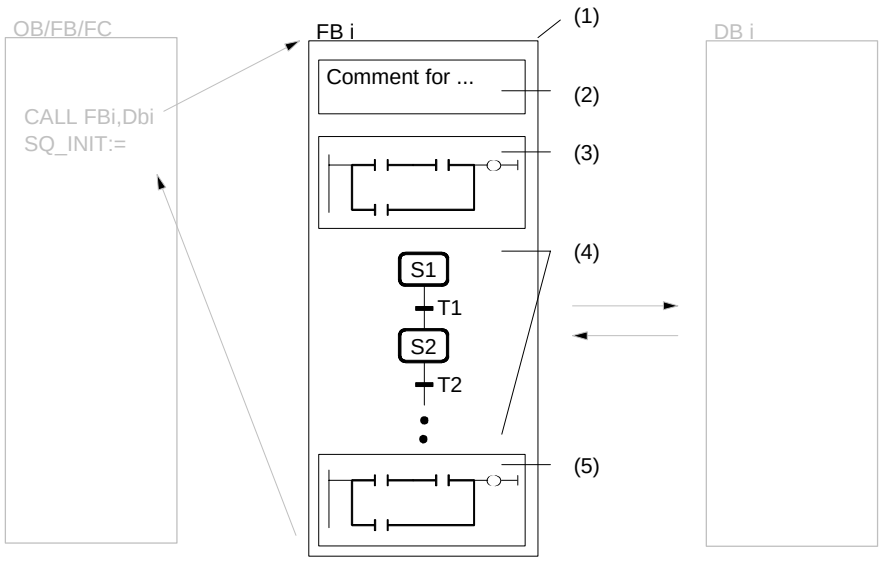
Opening in the S7-GRAPH Editor

In the S7-GRAPH editor, proceed as follows:

1. Select the menu command **File > Open** or click the corresponding button.
2. In the next dialog box, select the block or the source file you want to open. Decide whether you want to open a block offline on the PG or directly online at the CPU.

6 Programming the Structure of a Sequencer

The section below explains the rules and elements for creating a sequencer.



The heart of a sequential control system is made up of the S7-GRAPH FBs (1). These contain the following:

- The program for step by step execution of the controller in sequencers (4).
- Permanent instructions (3)(5) that are valid for all sequencers of the FB (and are displayed in a separate window)
- A block comment (2) with which you can describe the entire block.

How to Work with S7-GRAPH FBs

The order in which you perform the steps below is relatively unimportant. The order shown is simply a suggestion.

Step	Task
1	Specify the structure of the sequencer with the S7-GRAPH elements
2	Program the permanent instructions (optional)
3	Add a comment about the S7-GRAPH FB in the field for the block comment (optional)
4	Edit the steps and transitions created in point 1 individually

6.1 Sequencer - Principle

A sequencer consists of a series of steps and transitions that are activated in a fixed order depending on the step enabling conditions.

How a Sequencer is Executed

A sequencer is always executed starting with

- an initial step or
- several initial steps located at any position in the sequencer

As long as the actions of a step are being executed, this step **is active**. If several steps are being executed at the same time, they are all active steps.

↓

An active step is exited when

- any active disturbances have been eliminated or confirmed
- and the transition following the step is satisfied.

↓

The next step following the satisfied transition becomes **active**.

↓

At the end of a sequencer, there is

- a jump to any step in this sequencer or to another sequencer of the FB.
  This allows cyclic operation of the sequencer.
- a branch stop. The sequence of steps stops when the branch stop is reached.

Step

The control task is divided into single steps. Actions are formulated in the steps that are executed by the controller in a certain status (for example for controlling the outputs or activating and deactivating steps).

Active Step

An active step is a step whose actions are currently being executed.

A step is activated, as follows:

- When the conditions of the previous transition are satisfied or
- When the step is defined as an initial step and the sequencer has been initialized or
- When it is called by an event-dependent action.

Note

Avoid using a jump from a transition to an immediately preceding step.

To make this jump statement possible, insert an empty step and transition without conditions.

6.2 Structures of a Sequencer

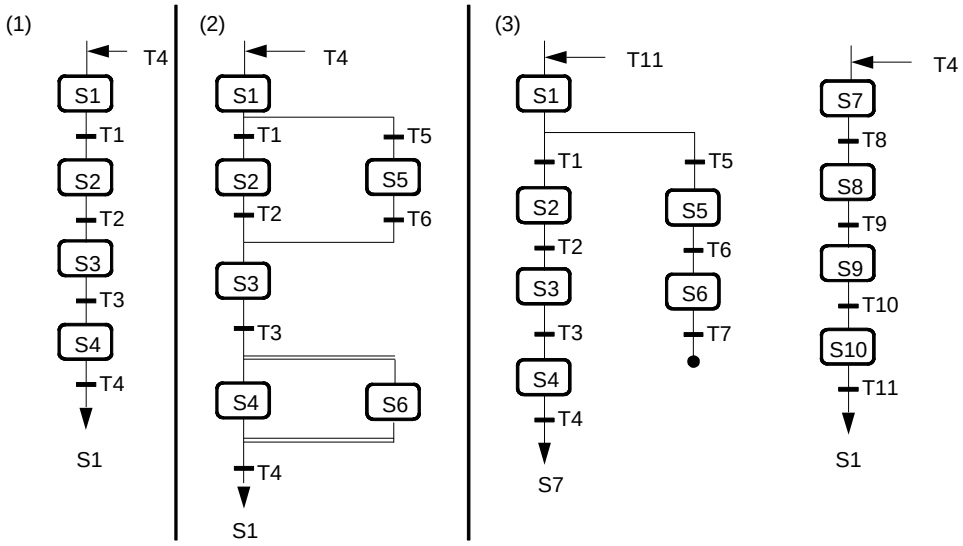
The simplest structure of a sequencer is a linear sequence of steps and transitions without branches.

A linear sequencer starts with a step and finishes with a transition that can be followed either by a jump to any step or by a branch stop.

The linear sequence can be extended by the following:

- Branches (alternative branch, simultaneous branch),
- Jumps to any steps,
- Further sequencers operating either dependent on the first sequencer or completely independently.
- Permanent instructions located before or after the sequencer.

Examples of Structures of a Sequencer



(1) S7 Graph FB with a linear sequencer

(2) S7 Graph FB with a sequencer with an alternative and simultaneous

(3) S7 Graph FB with two sequencers

6.3 Rules for the Structure of a Sequencer

The structure of the sequencer must satisfy the following rules:

- An S7-GRAPH FB can contain up to 250 steps or 250 transitions. Steps and transitions can only be inserted as pairs.
- When the S7-GRAPH FB is called, the sequencers are started
  - at the first step of the sequencer or
  - at an initial step.
- A sequencer can contain a maximum of 256 branches with
  - up to 125 alternative branches or
  - up to 249 simultaneous branches.

  It is not practical to generate more than 20 to 30 branches (depends on the CPU) due to their influence on program runtime.
- A branch can only be joined to a branch at the left of the branch that is to be closed.
- Jumps can be added following a transition at the end of a branch. These lead to a junction before a step in the same or in a different sequencer in the current FB.
- A sequencer stop can be added after a transition at the end of a branch and disables execution of this branch.
- Permanent instructions can be defined before or after the sequencer in the reserved field. These are called once in each cycle.

6.4 Elements of a Sequencer

The following list shows the elements that can make up a sequencer and the corresponding element buttons in the "Sequencer" toolbar with which you can create these elements:

 Step + Transition

 Jump

 Open Alternative Branch

 Close Alternative Branch

 Open Simultaneous Branch

 Close Simultaneous Branch

 Branch Stop

 Insert Sequencer

You program the structure of the sequencer at the "Sequencer" display level.

6.5 Step/Transition Pair and Initial Step

Step/Transition Pair

As default, the S7-GRAPH FB already contains one step/transition pair to which you can append further step/transition pairs.

Steps and transitions can only be inserted in pairs in a sequencer.

When you insert steps and transitions, they are automatically assigned a number. To ensure a better overview, you can renumber steps and transitions within a certain range later so that the numbering is consecutive.

Initial Step

The initial step is the step in a sequencer that becomes active without previously querying conditions when an S7-GRAPH FB is first called. The initial step is not necessarily the first step in a sequencer.

When a sequencer is run through cyclically, this step (just like every other step) only becomes active when the conditions of the previous transition are satisfied.

The sequencer is initialized by the FB parameter INIT_SQ = 1. In other words the sequencer starts with the initial step.

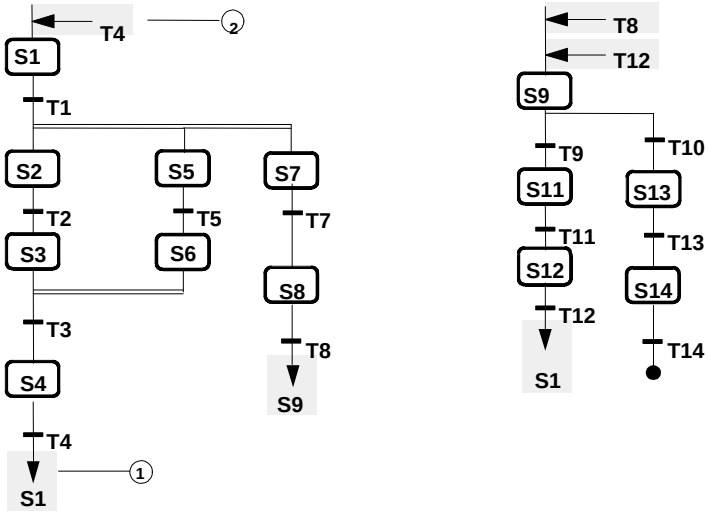
6.6 Jump

A jump is the change from a transition to any step within the sequencer or within another sequencer in the same FB.

A jump always follows a transition and closes the sequencer or the path of the branch at this point.

In contrast to a branch stop, the jump has the effect of repeating the execution of the sequencer or parts of it.

The jump (1) and the jump destination (2) are displayed graphically as an arrow but the connection itself is not visible.



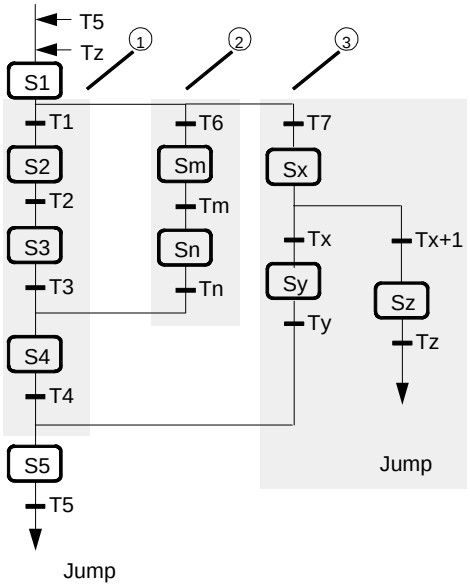
Caution

A jump to a simultaneous branch should be avoided.

6.7 Alternative Branch

An alternative branch consists of two or more parallel sub branches (a maximum of 125). Each path in an alternative branch begins with a transition. Only the branch path whose transition switches first is executed. An alternative branch therefore corresponds to an OR operation in which only one path can be active.

Each path in an alternative branch ends with a transition and can be closed by a branch stop or a jump.



①: Alternative sub-branch 1
②: Alternative sub-branch 2
③: Alternative sub-branch 3 with a further alternative branch

Priorities in Alternative Branches

If several transitions are satisfied at the same time at the start of various branches, the transition located furthest left has the highest priority.

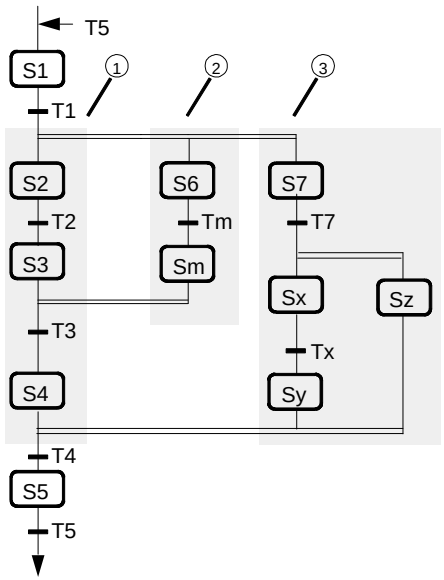
6.8 Simultaneous Branch

A simultaneous branch consists of more than one parallel path (a maximum of 249) each of which starts with a step. The path are executed simultaneously. A simultaneous branch corresponds to an AND branch.

The transition before the simultaneous branch (in the figure T1 and T7) activates the first steps of the individual simultaneous branch paths.

Every path in a simultaneous branch ends with a step and is completed by a successor transition.

If more than one simultaneous branch path joins to a transition, this transition only enables the next step when all active simultaneous branch paths have been executed.



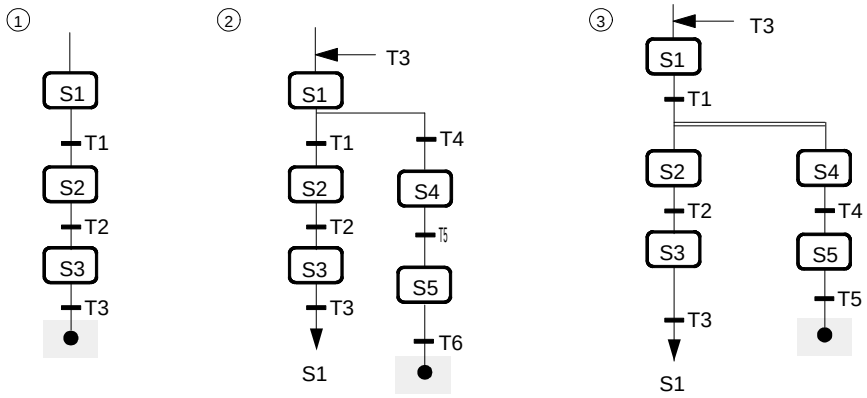
①: Simultaneous sub-branch 1
②: Simultaneous sub-branch 2
③: Simultaneous sub-branch 3 with a further simultaneous branch

6.9 Branch Stop

A branch stop at the end of a linear sequencer (1) or at the end of a path of an alternative branch (2) terminates the sequencer. The sequencer is not executed cyclically.

A branch stop at the end of a branch path of a simultaneous branch (3) terminates only this branch path. The execution of the other paths in the simultaneous branch continues.

A branch stop always follows a transition.



Caution

If all the paths of a sequencer are completed with a branch stop, the sequencer can only be restarted by the parameter INIT_SQ.

6.10 New Sequencer

An S7-GRAPH FB can consist of several sequencers.

Each sequencer can

1. be executed independent of the other sequencers and therefore at the same time.
2. be used with a jump to continue it or connect it as the branch of another sequencer. This allows you to represent technically complicated functions in one block providing you with a clearer structure for the program.

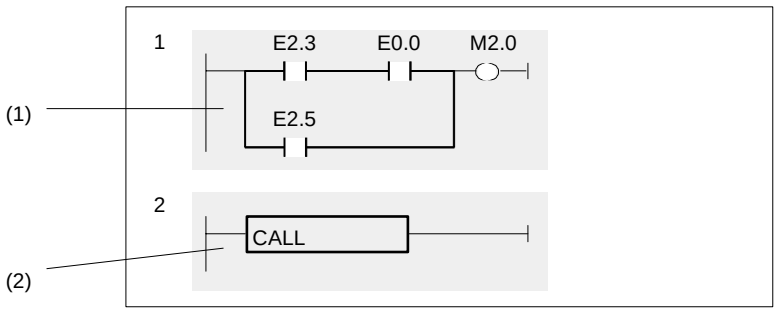
In every sequencer, an initial step can identify the first active step.

6.11 Permanent Instructions

Permanent instructions are conditions (1) and block calls (2) located before or after the sequencer. These are executed once per scan cycle regardless of the state of the sequencer.

You can program any number of permanent instructions in an S7-GRAPH FB. Each permanent instruction is assigned a number.

You program permanent instructions in the "permanent instructions" view.



Permanent Condition

Conditions that must be satisfied at more than one point in the sequencers can be programmed once centrally as permanent conditions. You can use the Ladder Logic elements: normally-open contact, normally-closed contact, and comparator or the FBD elements AND box, OR box, and comparator. A maximum of 32 Ladder Logic/FBD elements can be programmed per permanent condition.

The result of combining the conditions is stored in a ladder coil or in an FBD assignment with which the functions set and reset are available. The address used (for example a memory bit) can be queried in a transition, interlock, and supervision.

Permanent Block Call

Blocks that were created in a programming language other than S7-GRAPH can be called using permanent block calls in an S7-GRAPH FB or action. After the called block has been executed, the execution of the S7-GRAPH FB is continued.

When using a block call, note the following points:

- You can call the following blocks:
  - Functions (FC) and function blocks (FB) programmed in STL, LAD, FBD or SCL and
  - System functions (SFC) and system function blocks (SFB).
- Function blocks and system function blocks must be assigned an instance DB when they are called.
- The blocks must already exist in the "Blocks" folder of the S7 program before they are called.
- The names of the blocks can be specified in absolute form (for example FC1) or, if they are declared in the symbol table, symbolically (for example Motor 1).·
- When the blocks are called, you must supply the formal parameters of the called block with actual values.

6.12 Block Comment

Every S7-GRAPH FB can include a block comment. The block comment can be a maximum of 2048 characters long. It has no influence on the execution of the program.

Block comments are displayed only in the "Sequencer" view.

Entering the Block Comment

You can enter a block comment when the menu command **View > Display with > Comment** is selected (check mark).

1. Select the "Sequencer" display level.
2. Enter the block comment directly in the text box located above the sequencer.

7 Programming Conditions and Actions

The section below describes how to fill out the structure of the sequencers with content. You will learn about:

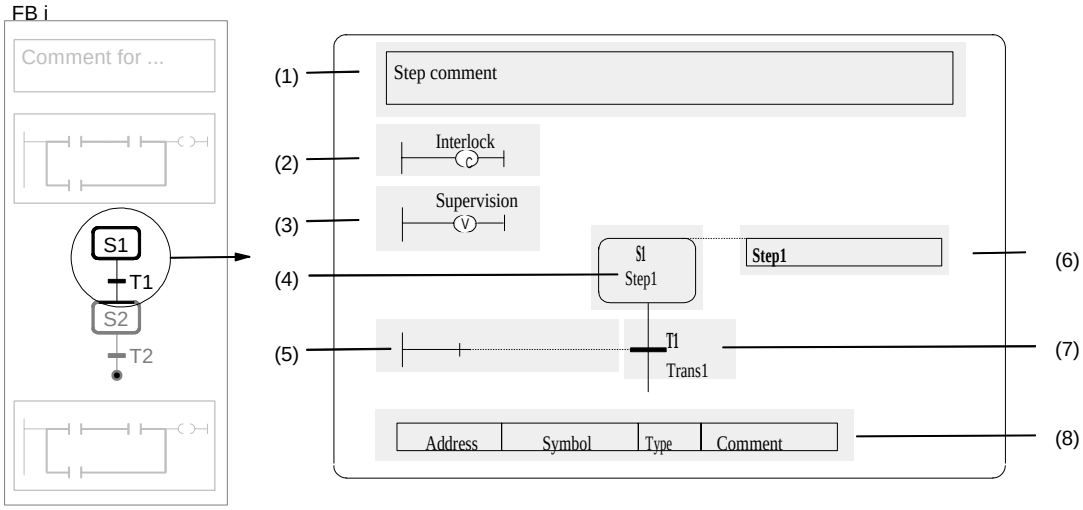
- Actions with their categories and components and how to make them dependent on events
- Conditions, where they occur and how to combine them.
- Ways of exiting a step when a supervision error has occurred.
- The use of symbolic programming in S7-GRAPH.

Opening blocks with the relevant editor:

The block used in the action table of a step can be opened with the relevant editor by selecting and double-clicking it. It is also possible to open the block with the relevant editor by selecting the block in the action table and then selecting the menu command **Edit > Open Block**.

Areas for Conditions and Actions

At the single step display level, almost all areas are displayed in which actions and conditions can be programmed.



(1) Field for the step comment
(2) Area for interlock conditions
(3) Area for supervision conditions
(4) Symbol for the step
(5) Area for transition conditions
(6) Empty action table
(7) Symbol for transition
(8) Symbol table for symbolic programming

Steps for Editing a Step/Transition Pair

Once you have programmed the structure of the sequencers in the S7-GRAPH FB, you can start to program the individual steps and transitions.

The order in which you work is of little importance. The order of the steps as shown below is simply a suggestion.

Step	Task
1	Programming the actions in the action table
2	Programming the conditions in transitions, interlocks, supervisions and the permanent conditions
3	Editing the numbers and names of steps and transitions or entering a step comment
4	Editing the section of the STEP 7 symbol table for this step

7.1 Actions

In the steps, you program actions that, for example, control the inputs, outputs and memory bits, activate or deactivate the steps of the sequencer or call blocks. This means that actions contain instructions for process control. These actions are executed in order from "top" to "bottom" when the step is active.

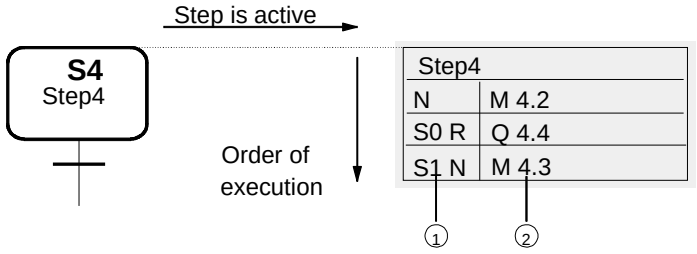
You program the step contents in the action table at the "single step" display level. You can also select the "Sequencer" display level and display the actions with the menu command **View > Display With > Conditions and Actions**.

Components of an Action

An action consists of the following:

(1): An optional event (for example S1) and an instruction (for example N) and

(2): An address (for example M4.3) or an assignment (for example A:=B+C).



Categories of Actions

Actions are divided into the following categories:

- Standard actions
  - With and without interlock
- Event-dependent actions
  - With and without interlock
  - To activate and deactivate steps
- Counters, timers and arithmetic in actions

Empty Steps (Steps Without Actions)

Steps that do not contain programmed actions are empty steps. An empty step behaves like an active step. The successor transition becomes valid.

7.2 Standard Actions

Standard Actions with and without Interlock

All standard actions can be combined with an interlock. The actions are only executed when the conditions of the interlock are satisfied (letter C added).

Standard actions without an interlock are executed as long as the step is active.

Instruction	Address Identifier	Address Location	Meaning:
N[C]	Q,I,M,D	m.n	As long as the step is active [and interlock satisfied], the address is set to 1.
S[C]	Q,I,M,D	m.n	**S**et: As soon as the step is active [and the interlock is satisfied], the address is set to 1 and then remains set to 1 (latching)
R[C]	Q,I,M,D	m.n	**R**eset: As soon as the step is active [and the interlock is satisfied], the address is set to 0 and remains at 0 (latching).
D[C]	Q,I,M,D	m.n	On (**D**elay): n seconds after the step is activated, the address has signal 1 [if the interlock is satisfied] while the step is active. This does not apply if the step is active for a time shorter than n seconds (non-latching).
	T#<const>		Time constant
L[C]	Q,I,M,D	m.n	**L**imited pulse: If the step is active [and the interlock is satisfied], the address is set to 1 for n seconds (non latching).
	T#<const>		Time constant
CALL[C]	FB, FC, SFB, SFC	Block number	Block call: As long as the step is active [and the interlock is satisfied], the specified block is called.

[ ] = optional addition to the interlock; m = byte address; n = bit address; address range: 0.0 to 65535.7

Address D: DBi.DBXm.n = data block number i; DBXm.n = bit in the DB;

SFB/FB = System function block/ function block; SFC/FC = System function/function

For CALL[C] FB/SFB, an instance DB is required.

Note

You can open the called block or its instance DB using the **Open Block** function or by right-clicking or double-clicking on the block.

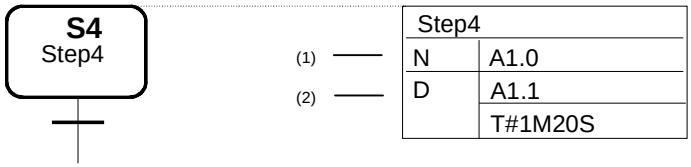
Time Constant

All actions that contain the instructions D or L require a time to be specified. Times are programmed as constants with the syntax **T#<const>** and can be combined as required.

<const>= nD (n days), nH (n hours), nM (n minutes), nS (n seconds), nMS (n milliseconds), where n = a number (integer)

Example: T#2D3H: time constant = 2 days and 3 hours

Example: Action Table with Standard Actions



(1): As long as the step is active, the signal at output Q1.0 is set to 1.

(2): 1 minute and 20 seconds after step activation and providing the step is still active, the signal at output Q1.1 is set to 1. If the step is no longer active, the signal at Q1.1 is 0.

7.3 Event-Dependent Actions

Events

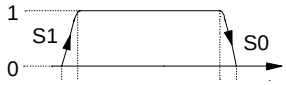
An action can be logically combined with an event. An event is the change in the signal state of a step, a supervision or an interlock or the acknowledgment of a message or a registration is set.

If an action is logically combined with an event, the signal state of the event is recognized by edge detection. This means that the instructions can only be executed in the cycle in which the event took place.

Events and Signal Evaluation

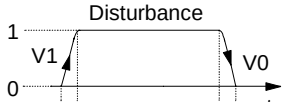
Step

- **S1**: Step becomes active
- **S0**: Step is deactivated



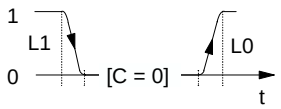
Supervision

- **V1**: Supervision error occurred (disturbance)
- **V0**: Supervision error cleared (no disturbance)



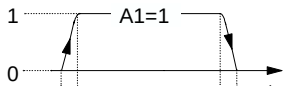
Interlock

- **L0**: Interlock condition entering state
- **L1**: Interlock condition leaving state (for example disturbance)
- **C**: Interlock condition is satisfied



Message and Registration

- **A1**: A message is acknowledged
- **R1**: A registration is set
  (rising edge at input REG_EF / REG_S)



Event-Dependent Actions - with and without Interlock

All standard actions with the exception of actions containing the D and L instructions can be logically combined with an event.

Event	Instruction	Address Identifier	Address Location	Meaning:
S1, V1, A1, R1	N[C], R[C], S[C] CALL[C]	Q,I,M,D, FB, FC, SFB, SFC	m.n block number	After the event is detected [and the interlock is active], the following applies to the next step cycle: • Address has signal state 1 (N) once • Address set to signal state 1 once (S) • Address set to signal state 0 (R) once • Block is called once (CALL)
S0, V0, L0, L1	N, R, S, CALL	Q,I,M,D FB, FC, SFB, SFC	m.n block number	After the event is detected, the following applies to the next step cycle: • Address has signal state 1 (N) once • Address set to signal state 0 (S) once • Address set to signal state 0 (R) once • Block is called once (CALL)

m = byte address; n = bit address; address range: 0.0 to 65535.7

Address D: DBi.DBXm.n = data block number i; DBXm.n = bit in the DB;

SFBi / FBi = (system) function block number i; SFCi / FCi = (system) function number i

Event-Dependent Actions - to Activate and Deactivate Steps

Other steps can be activated or deactivated using the ON and OFF instructions. The instructions are always dependent on a step event, in other words the event determines the time at which the step is activated or deactivated.

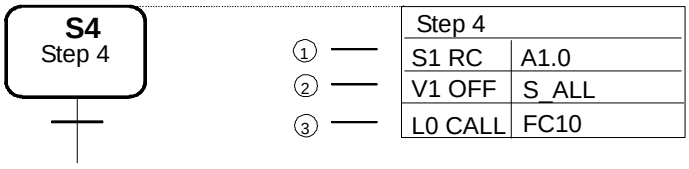
These instructions can be combined with an interlock. The actions are only executed when the conditions of the interlock are satisfied.

Event	Instruction	Address Identifier	Address Location	Meaning:
S1, V1, A1, R1	ON[C], OFF[C]	S	i	Activate (ON) or deactivate (OFF) the step dependent on the event [and interlock]
S1, V1	OFF[C]	S_ALL		Deactivate all steps dependent on the event [and interlock]. The exception to this is the step in which the action is located.
S0, V0, L0, L1	ON, OFF	S	i	Activate (ON) or deactivate (OFF) dependent on the event
L1	OFF	S_ALL		Deactivate all steps dependent on the event. The exception to this is the step in which the action is located.
i: step number				

Note

If a step is both activated and deactivated in one cycle, deactivating has higher priority.

Example: Action Table with Event-Dependent Actions



(1): As soon as the step is active and the interlock is satisfied, output Q1.0 is set to 0 and remains set to 0.

(2): As soon as the supervision error occurs, all active steps are deactivated, except for step S4 in which the action is located.

(3): As soon as the interlock condition is satisfied, the specified block (FC10) is called.

7.4 Counters, Timers, and Arithmetic in Actions

Counters in Actions

All counters in actions are always dependent on an event, in other words the event determines the time at which the instruction is activated. You also have the option of combining counters with an interlock. The count instructions combined with the interlock are only executed when the conditions of the interlock are satisfied (letter C added) as well as the event occurring. Counters without an interlock are executed when the event occurs.

The counters in actions behave like the counters in other S7 programming languages: There is no overflow and no underflow and with a counter reading of 0, the counter bit is also 0, otherwise it is 1.

Event	Instruction	Address Identifier	Address Location	Meaning:
S1, S0, L1, L0, V1, V0, A1, R1	CS[C]	C	X	Set: As soon as the event occurs [and the interlock is satisfied], the initial counter value is loaded in the counter.
		<initial counter value>		Initial counter value
S1, S0, L1, L0, V1, V0, A1, R1	CU[C]	C	X	Count up: As soon as the event occurs [and the interlock is satisfied], the counter value is incremented by "1".
S1, S0, L1, L0, V1, V0, A1, R1	CD[C]	C	X	Count down: As soon as the event occurs [and the interlock is satisfied], the counter value is decremented by "1".
S1, S0, L1, L0, V1, V0, A1, R1	CR[C]	C	X	Reset: As soon as the event occurs [and the interlock is satisfied], the counter value is reset to 0.

[ ] = optional supplement to interlock; x = number of the counter

Initial Counter Value

All actions containing the CS instruction require an initial counter value. The initial counter value is programmed with the following syntax:

<initial counter value> = IWy, QWy, MWy, LWy, DBWy, DIWy; variable of type WORD; C#0 ... C#999

Y = 0 ... 65534

Timers in Actions

All timers in actions are always dependent on an event, in other words the event determines the time at which the instruction is activated. You also have the option of combining timers with an interlock. The timer instructions combined with an interlock are only executed when the conditions of the interlock are satisfied in addition to the event occurring (additional letter C). Timers without an interlock are executed when the event occurs.

Event	Instruction	Address Identifier	Address Location	Meaning:
S1, S0, L1, L0, V1, V0, A1, R1	TL[C]	T	X	Extended pulse, non-latching behavior of the timer bit (status of the timer): As soon as the event occurs, the timer starts. The condition C is only relevant for the point at which the timer starts. Once the timer starts, it continues to run regardless of the interlock and step activity. During the period <time>, the timer bit is set to "1", afterwards it is reset to "0". The started timer can be restarted by an event occurring. The timer bit remains set to "1", the timer starts again with the value of <time>.
		<Time>		time
S1, S0, L1, L0, V1, V0, A1, R1	TD[C]	T	X	Latching on delay, latching behavior of the timer bit (status of the timer): As soon as the event occurs, the timer starts. The condition C is only relevant for the point at which the timer starts. Once the timer starts, it continues to run regardless of the interlock and step activity. During the <time>, the timer bit is set to "0". The started timer can be restarted by an event occurring. The timer bit remains set to "0", the total time the timer runs is extended by <time>. The timer bit is set to "1" only after the total time has elapsed.
		<Time>		time
S1, S0, L1, L0, V1, V0, A1, R1	TR[C]	T	X	Reset: As soon as the event occurs [and the interlock is satisfied], the timer is stopped. Timer bit (status of the timer) and timer value are reset to "0".

[ ] = optional supplement to interlock; x = number of the timer

Time (duration)

All actions containing one of the TL or TD instructions require a time (duration) to be specified. The time is programmed with the following syntax:

<time> = IWy, QWy, MWy, LWy, DBWy, DIWy; variable of the type S5TIME, WORD; S5T#time_cnstant

Y = 0 ... 65534

Arithmetic in Actions

You can provide statements with simple arithmetic expressions in actions. There are assignments in the form A:=B, A:=func(B) and A:=B<operator>C. An action containing an arithmetic expression requires the N instruction. The action can be dependent on an event. This means that the assignment can either be executed only when the relevant event occurs or in each cycle in which the step is active. As an option, the action can also be combined with an interlock (letter C added).

Event	Instruction	Assignment	Meaning
--	N[C]	A:=B A:=func(B) A:=B<operator>C	As long as the step is active [and the interlock is satisfied], the assignment is executed.
S0, S1, V0, V1, L0, L1, A1, R1	N[C]	A:=B A:=func(B) A:=B<operator>C	After the event occurs [and the interlock is true], the assignment is executed.

Direct Assignments

You can enter direct assignments with the syntax **A:=B**. The following data types can be used for a direct assignment:

- 8 bits: BYTE, CHAR
- 16 bits: WORD, INT, DATE, S5TIME
- 32 bits: DWORD, DINT, REAL, TIME, TIME_OF_DAY

The following table is an overview of all the permitted types of assignment. You can assign not only addresses of the same type but also addresses with different data types with the same length and addresses of different lengths to each other:

		8 Bits		16 Bits				32 Bits				
		BYTE	CHAR	WORD	INT	DATE	S5 TIME	DWORD	DINT	REAL	TIME	TIME_OF _DAY
8 Bits	BYTE :=	✓	✓									
	CHAR :=	✓	✓									
16 Bits	WORD :=	✓	✓	✓	✓	✓	✓					
	INT :=	✓	✓	✓	✓	✓						
	DATE :=	✓		✓	✓	✓						
	S5TIME :=			✓			✓					
32 Bits	DWORD :=	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	DINT :=							✓	✓		✓	✓
	REAL :=							✓		✓		
	TIME :=	✓		✓	✓			✓	✓		✓	
	TIME_OF_ DAY :=	✓		✓	✓			✓	✓			✓

Assignments with Built-in Function

Assignments with a built-in function are entered with the syntax **A:=func(B)**. These assignments are essentially conversions functions and complex math functions. The assigned address A specifies the data type of the expression. You can use the following assignments with a built-in function:

Assignment with Built-in Function	**Comment**
Conversion functions:	
A := BCD_TO_NUM(B)	BCD to INT or DINT (STL instructions: BTI, BTD)
A := NUM_TO_BCD(B)	INT or DINT to BCD (STL instructions: ITB, DTB)
A := INT_TO_DINT(B)	INT to DINT (STL instruction: ITD)
A := DINT_TO_REAL(B)	DINT to REAL (STL instruction: DTR)
A := ROUND(B)	REAL to DINT (STL instruction: RND)
A := TRUNC(B)	REAL to DINT, truncate remainder (STL instruction: TRUNC)

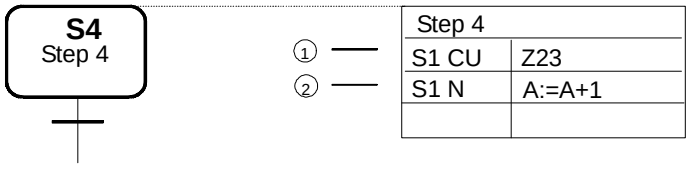
Assignment with Built-in Function	**Comment**
Math functions:	
A := NEGR(B)	REAL negation
A := ABS(B)	REAL absolute value
A := SQR(B)	REAL square
A := SQRT(B)	REAL square root
A := LN(B)	REAL logarithm to base e
A := EXP(B)	REAL exponent to base e
A := SIN(B)	REAL sine
A := ASIN(B)	REAL arc sine
A := COS(B)	REAL cosine
A := ACOS(B)	REAL arc cosine
A := TAN(B)	REAL tangent
A := ATAN(B)	REAL arc tangent
A := NEG(B)	Negation (two's complement) (STL instructions: NEGI, NEGD, NEGR)
Other Functions:	
A := NOT(B)	One's complement (STL instructions: INVI, INVD)
A := SWAP(B)	Swap bytes (STL instructions: TAD, TAW)
A := RLDA(B)	Rotate 32 bits left by 1 bit via CC1 (STL instruction: RLDA)
A := RRDA(B)	Rotate 32 bits right by 1 bit via CC1 (STL instruction: RRDA)

Assignments with Operator

Enter assignments with an operator with the syntax **A:=B<operator>C**. These assignments are basic math functions and bit logic operations. The assigned address A specifies the data type of the expression. You can use the following assignments with an operator:

Assignments with Operator	Comment
A := B + C	+I, +D, +R
A := B - C	-I, -D, -R
A := B ∗ C	∗I, ∗D, ∗R
A := B / C	/I, /D, /R When necessary, the operator can be supplemented by +D/+R in the basic math functions.
A := B MOD C	Modulo: Only for data type DINT
A := B AND C	AND operation (STL instructions: AW, AD)
A := B OR C	OR operation (STL instructions: OW, OD)
A := B XOR C	EXCLUSIVE OR operation (STL instructions: XOW, XOD)
A := B SHL C	Shift left, 0<=C<=255 (STL instructions: SLW, SLD)
A := B SHR C	Shift right, 0<=C<=255 (STL instructions: SRW, SRD)
A := B SSR C	Shift right with sign, 0<=C<=255 (STL instructions: SSI, SSD)
A := B ROL C	Rotate left, 0<=C<=255 (STL instructions: RLD)
A := B ROR C	Rotate right, 0<=C<=255 (STL instructions: RRD)

Example: Action Table with Counters and Arithmetic



Two alternatives for counting production cycles:

(1): As soon as the step is active, the counter C23 is incremented by 1. The counter therefore counts the number of times the step is activated.

(2): As soon as the step becomes active, the value of A is incremented by 1. In this way, the arithmetic can be used to count actions based on the number of activations of the step.

7.5 Conditions

Conditions are binary states of the process that are combined with each other as LAD or FBD elements (normally-open contact, normally-closed contact, AND box, OR box, comparator) in the LAD diagram or in the function block diagram according to Boolean logic. The result of the logic operation (RLO) can influence individual actions of a step, the entire step, the enabling of the next step or the entire sequencer. Logic operations are programmed in Ladder Logic or FBD.

Conditions are :

- Events (for example active step is exited) and
- States (for example input I2.1 is set)

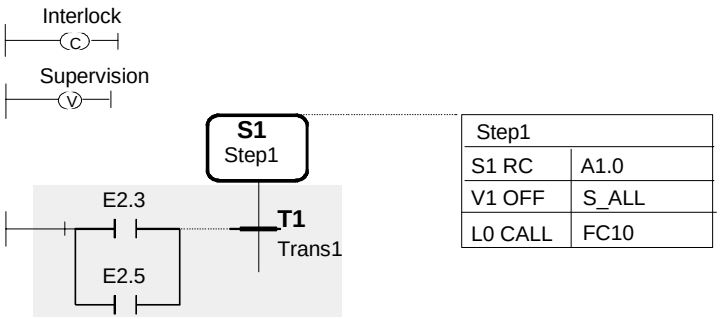
Areas in which Conditions Can Occur

Conditions occur in the following areas:

- Transition (enabling the next step)
- Interlock
- Supervision
- Permanent instructions (conditions and block calls at the start and/or at the end of the sequencer)

Transition

In a transition, you program conditions that control how the sequencer passes control from one step to the next.



A transition is displayed and programmed at the single step display level. You can also select the "Sequencer" display level and display the transitions with the menu command **View > Display With > Conditions and Actions**.

The transition passes control to the next step of the sequencer when the logic operation of the conditions is satisfied, in other words when the network produces the result 1. The step following the transition becomes active.

The transition does not pass control to the next step of the sequencer if the logic operation of the conditions is not satisfied; in other words, when the network produces the result 0. The active step remains active.

Empty Transition

Transitions without programmed conditions are empty transitions. This does not affect the execution of the sequencer. The active step is run through only once. The following step becomes valid.

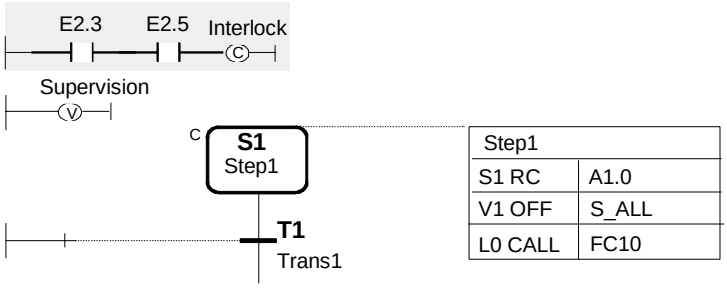
Skipping Steps

If both the transition before a step and the transition after a step are valid at the same time, the step does not become active.

In the "Compile/Save" tab of the "Block Settings" dialog box (menu command **Options > Block Settings**), the option "Skip Steps" must be selected.

Interlock

An interlock is a programmable condition for interlocking the step that influences the execution of individual actions.



If the logic operation of the conditions is satisfied, the actions combined with interlocks are executed.

If the logic operation of the conditions is not satisfied, there is a disturbance:

- Actions combined with an interlock are not executed
- An interlock error is signaled (event L1).

You program an interlock at the single step display level. A maximum of 32 Ladder Logic/FBD elements can be programmed per interlock. The result of the operation is managed automatically by the system.

A programmed interlock is indicated by the letter C following the step at all display levels.

Caution

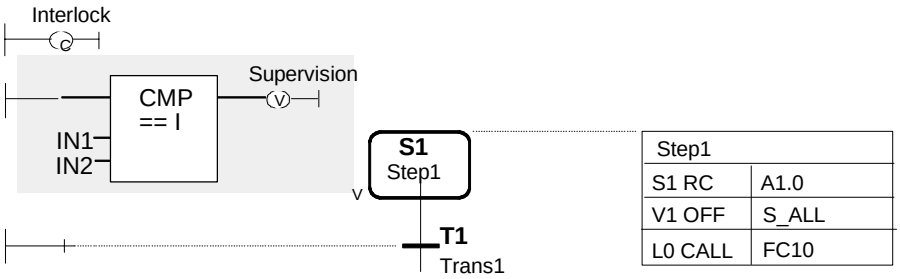
If you program an interlock this is only used in the actions when their instructions have been extended by the letter C.

Empty Interlock

An interlock without a condition (in other words without Ladder Logic or FBD elements) behaves like a satisfied interlock.

Supervision

A supervision is a programmable condition for monitoring a step that influences the way in which the sequencer passes control from one step to the next.



A programmed supervision is indicated by the letter V to the left of the step at all display levels.

You program supervisions at the single step display level. A maximum of 32 Ladder Logic/FBD elements can be programmed per supervision. The result is managed automatically by the system.

If the logic operation of the conditions is satisfied, there is a disturbance and the event V1 is signaled. The sequencer does not pass control to the next step. The current step, however, remains active. The step activation time Si.U is stopped as soon as the condition is satisfied.

If the logic operation of the conditions is not satisfied, there is no disturbance. If the successor transition is satisfied, the sequencer passes control to the next step.

Permanent Instructions

You program permanent instructions with the familiar Ladder Logic/FBD elements. For more detailed information, refer to the "Permanent Instructions" section in "Programming the Structure of a Sequencer".

7.6 Ladder Logic Elements for Programming Conditions

Ladder Logic Elements

A ladder logic program follows the power flow between power rails. The individual ladder elements carry binary information: Signal state "0" (current not flowing) or "1" (current flowing).

Ladder Logic Element	Address	Data type	Memory Area	Description
Normally open contact <Address> 	**<Operand>**	BOOL, TIMER, COUNTER	I, Q, M, T, C data bit	The normally open contact has signal state "1", when the signal state of the address is "1". The address specifies the bit whose signal state is queried.
Normally closed contact <Address> 	**<Operand>**	BOOL, TIMER, COUNTER	I, Q, M, T, C data bit	The normally closed contact has signal state "1", when the signal state of the address "0". The address specifies the bit whose signal state is queried.
Coil (output) <Operand> -----(=)	**<Operand>**	BOOL	I, Q, M, data bit	You use coils only in the permanent instructions of a sequencer. The address contains the result of the conditions that were programmed within the permanent instructions (logic operations with normally open contacts, normally closed contacts, compare boxes). It specifies the bit to which the signal state is assigned.
Coil **Set output** <Operand> -----(S)	**<Operand>**	BOOL	I, Q, M, data bit	"Set coil" is only executed when the result of logic operation (RLO) of the previous permanent instructions is "1". The <address> is then set to "1". An RLO of "0" has no effect, so the current signal state of the address is not changed. The address specifies which bit will be set.
Coil **Reset output** <Operand> -----(R)	**<Operand>**	BOOL	I, Q, M, data bit	"Reset coil" is only executed when the result of logic operation (RLO) of the previous permanent instructions is "1". The <address> is then set to "0". An RLO of "0" has no effect, so the current signal state of the address is not changed. The address specifies which bit will be reset.

Rules for Entering Ladder Logic Elements

When entering Ladder elements, keep to the following rules:

- All elements and branches must be connected together.
- Each branch in a Ladder Logic network must contain at least one Ladder Logic element.

Modes for Inserting Ladder Elements

There are two editing modes available for inserting Ladder elements:

- **Insert > Direct** In this mode, you first select the position in the sequencer at which you want to insert an element and then select the element to be inserted.
- **Insert > Drag and Drop** In this mode, you first select the element you want to insert and then select the position at which the element will be inserted using the mouse pointer. This mode is particularly suitable when you want to insert the same element at more than one position in the sequencer.

Entering Addresses

There are two ways of entering addresses:

- Click the placeholder "??.?" of the required text field. Then enter an absolute or symbolic address (for example I1.0, "limit_switch"). You can use both the German mnemonics (E, A) and the English mnemonics (I, O) for addresses. Set the mnemonics you require in the SIMATIC Manager.
- Select the address and then select an absolute or symbolic address from the drop-down list box (for example I1.0, "Limit switch"). The address is then entered.

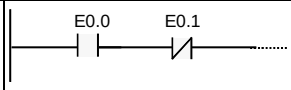
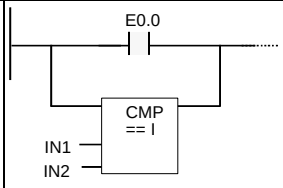
Changing Over Between Ladder Logic and FBD

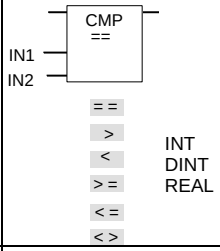
In S7-GRAPH, you can program conditions in one of two programming languages: In ladder diagram (LAD) or in function block diagram (FBD). You can display the conditions in LAD or in FBD at any time. To toggle the display, simply select the appropriate menu command: **View > LAD** or **View > FBD**.

7.7 Logic Operations with Ladder Logic Elements

A logic operation queries the signal state of a Ladder element for "0" (not activated, off) or "1" (activated, on) and then supplies a result of logic operation (RLO). The instruction either saves the result or executes a Boolean logic operation with it.

The two logic operations AND and OR must be distinguished:

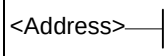
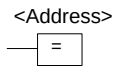
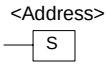
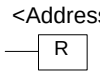
Logic operation	Example of a logic operation in	a transition
AND		The condition is satisfied when the signal state at input I '0.0 is "1" AND and the signal state at input I 0.1 is "0".
OR		The condition is satisfied when • The signal state at input I 0.0 is "1" OR • The compare box produces the result “true" after comparing the two addresses.

Ladder Logic Element	**Addresses**	**Data**	**Memory Area**	**Description**
Compararator				
	IN1: First comparison value	INT/ DINT/ REAL/ TIMER/ CONST	I, Q, M, data (double) word	The compare box has the signal state "1" when the comparison between the two addresses IN1 and IN2 is true. INT: 16-bit integer comparator Parameter: words DINT: 32-bit integer comparator and comparator for timers Parameter: double words, timers REAL: 32-bit floating-point number comparator Parameter: double words
	IN2: Second comparison value	INT / DINT / REAL / TIMER / CONST	I, Q, M, data (double) word	**Note:** When you enter the comparator you only need to specify the data type D/R/I explicitly when you enter the 32 bit values without specifying a type (for example when accessing absolute addresses in the double word area). Otherwise the data type is detected by the system so that the specification can be omitted.

7.8 FBD Elements for Programming Conditions

FBD Elements

An FBD program follows the signal flow of a logic string. The individual FBD elements carry binary information: Signal state "0" (current not flowing) or "1" (current flowing).

FBD Element	Address	Data type	Memory Area	Description
Insert binary input 	**<Operand>**	BOOL	I, Q, M, T, C, D, L	The instruction adds a further binary input to an AND or OR box after the position selected. The address specifies the bit whose signal state is queried.
Negate binary input 	**None:**	--	--	The instruction negates the result of logic operation (RLO).
Output				You use output operations only in the permanent instructions of a sequencer.
Assign output 	**<Operand>**	BOOL	I, Q, M, D, L	The address contains the result of the conditions that were programmed within the permanent instructions (logic operations before the output box). It specifies the bit to which the signal state is assigned.
Set output 	**<Operand>**	BOOL	I, Q, M, D, L	This instruction is executed only when the RLO of the previous permanent instruction is "1". The <address> is then set to "1". An RLO of "0" has no effect, so the current signal state of the address is not changed. The address specifies which bit will be set.
Reset output 	**<Operand>**	BOOL, TIMER, COUNTER	I, Q, M, T, C, D, L	This instruction is executed only when the RLO of the previous permanent instruction is "1". The <address> is then reset to "0". An RLO of "0" has no effect, so the current signal state of the address is not changed. The address specifies which bit will be reset.

Rules for Entering FBD Elements

When entering FBD elements, keep to the following rules:

- All elements and branches must be connected together.
- Each branch in an FBD network must contain at least one FBD element.

Modes for Inserting FBD Elements

There are two editing modes available for inserting FBD elements:

- **Insert > Direct** In this mode, you first select the position in the sequencer at which you want to insert an element and then select the element to be inserted.
- **Insert > Drag and Drop** In this mode, you first select the element you want to insert and then select the position at which the element will be inserted using the mouse pointer. This mode is particularly suitable when you want to insert the same element at more than one position in the sequencer.

Entering Addresses

There are two ways of entering addresses:

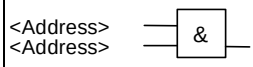
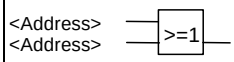
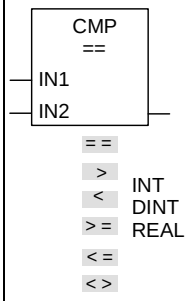
- Click the placeholder "??.?" of the required text field. Then enter an absolute or symbolic address (for example I1.0, "limit_switch"). You can use both the German mnemonics (E, A) and the English mnemonics (I, O) for addresses. Set the mnemonics you require in the SIMATIC Manager.
- Select the address and then select an absolute or symbolic address from the drop-down list box (for example I1.0, "Limit switch"). The address is then entered.

Changing Over Between FBD and Ladder Logic Diagram

In S7-GRAPH, you can program conditions in one of two programming languages: In ladder diagram (LAD) or in function block diagram (FBD). You can display the conditions in LAD or in FBD at any time. To toggle the display, simply select the appropriate menu command: **View > LAD** or **View > FBD**.

7.9 Logic Operations with FBD Elements

The two logic operations AND and OR must be distinguished: Here, two FBD elements are available.

FBD Element	Address	Data type	Memory Area	Description
AND operation 	**<Operand>**	BOOL	I, Q, M, T, C, D, L	The instruction provides the signal "1" when the signal state of all <addresses> is "1". If the signal state of an <address> is "0", the instruction produces the result "0". The address specifies the bit whose signal state is queried.
OR operation 	**<Operand>**	BOOL	I, Q, M, T, C, D, L	The instruction provides the signal "1" when the signal state of one of the <addresses> is "1". If the signal state of all <address> is "0", the instruction produces the result "0". The address specifies the bit whose signal state is queried.
Compare box 	**IN1**: First comparison value	INT/ DINT/ REAL	I, Q, M, D, L, constant	The result of the logic operation has the signal state "1" when the comparison between the two addresses IN1 and IN2 is true. INT: 16-bit integer comparison : Parameter words DINT: 32-bit integer comparison : Parameter double words REAL: 32-bit floating-point number comparison Parameter: double words
	IN2: Second comparison value	INT / DINT / REAL	I, Q, M, D, L, constant	**Note:** When you enter the comparator you only need to specify the data type D/R/I explicitly when you enter the 32 bit values without specifying a type (for example when accessing absolute addresses in the double word area). Otherwise the data type is detected by the system so that the specification can be omitted.

7.10 S7-GRAPH Addresses in Conditions

It is possible to use system information about steps as addresses in transitions, supervisions, interlocks, actions, and permanent instructions.

Address	Meaning	Used in
Si.T	Current or last activation time of step i	Comparator, assignment
Si.U	Total activation time of step i without the time of a disturbance	Comparator, assignment
Si.X	Indicates whether or not step i is active	Normally open contact/normally closed contact
Transi.TT	Transition i is satisfied Check whether all conditions of transition i are satisfied	Normally open contact/normally closed contact

You can use S7-GRAPH-specific addresses just like all other addresses in LAD and FBD or in actions.

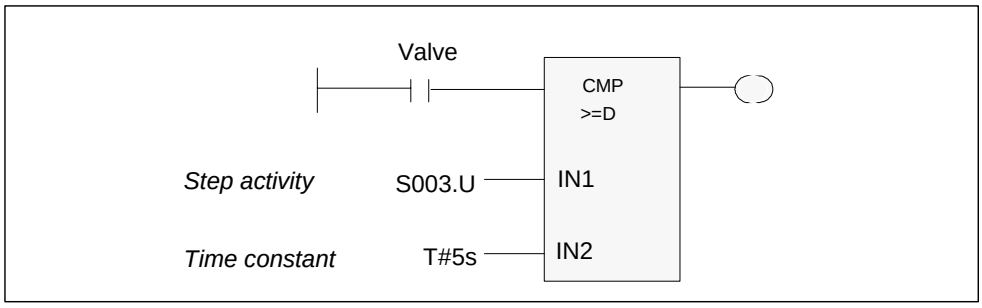
Example of Using S7-GRAPH Addresses

In many processes, it is necessary to monitor the duration of an activity (minus any disturbance times). Example: A product must be stirred for a total of 5 seconds regardless of the times in which the step was disturbed.

In situations such as this, you can program a supervision condition in which you monitor the Si.U address.

The "Compare integers (32-bit)" instruction allows you to compare the value of the Si.U address with the set time of 5 seconds. Inputs IN1 and IN2 are compared.

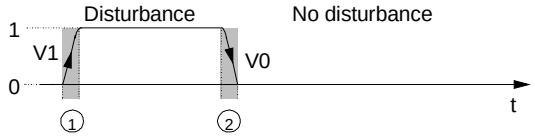
In the example, the undisturbed activation time of step 3 is compared with a selected time of 5 seconds. If the activation time of step 3 is greater than or equal to the selected time of 5 seconds, the condition is satisfied. If the valve is also closed (signal = 1), the supervision is satisfied.



7.11 Supervision Errors and Acknowledgment

Every step can be assigned a supervision condition. Only active step are monitored. S7-GRAPH recognizes whether or not the supervision conditions are satisfied (in other words when a supervision error has occurred).

A supervision error entering the state (1) results in event "V1", a supervision error leaving the state (2) results in the event "V0".



Disturbance: conditions in the supervision are satisfied
No disturbance: conditions in the supervision are not satisfied

Signaling and Acknowledgment - Requirements

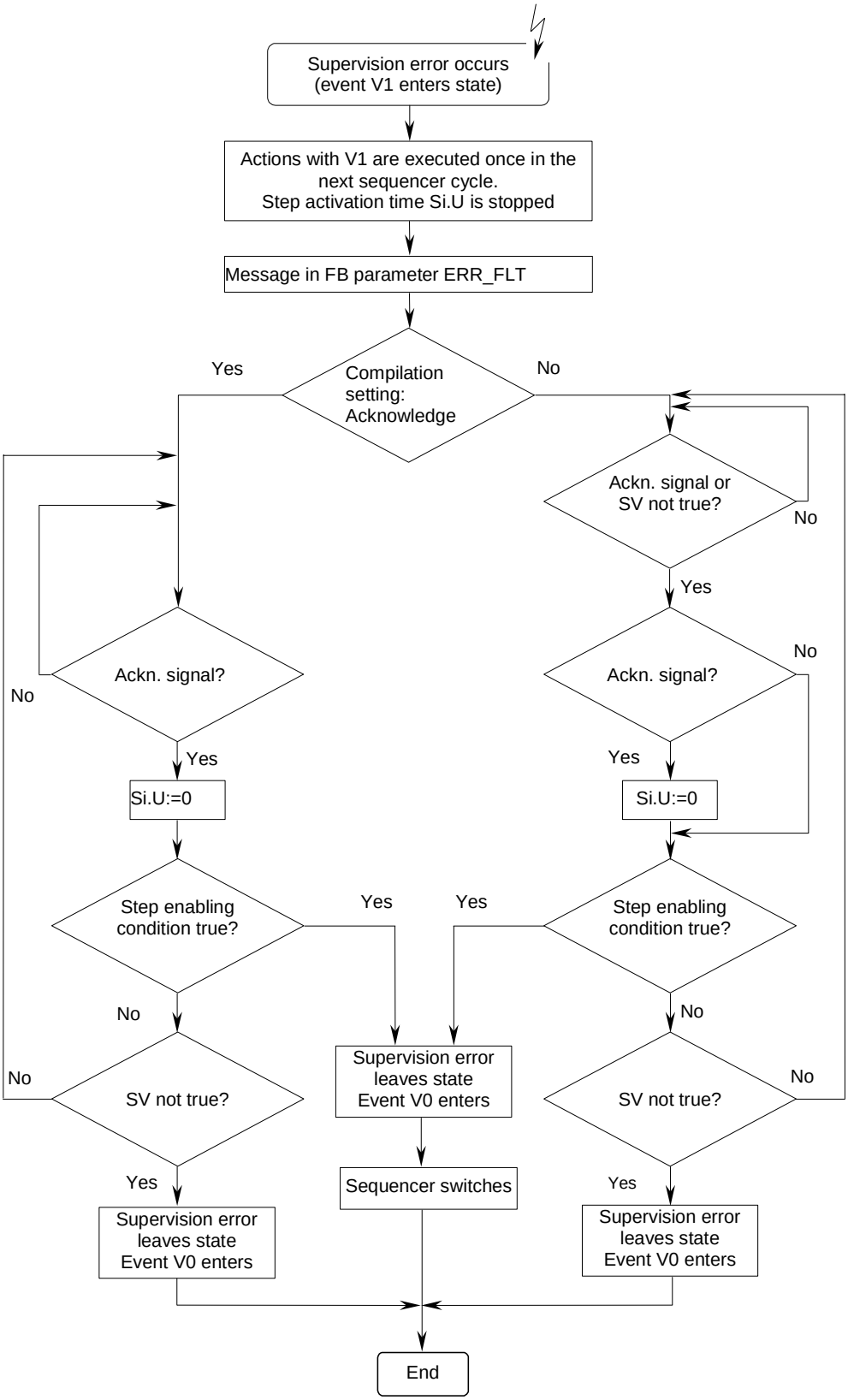
The signaling and acknowledgment of a supervision error is only possible with the following settings in the "Compile / Save" tab of the "Block Settings" dialog (menu command **Options > Block Settings**):

- "Standard", "Maximum", or "Definable" is set in the "FB Parameters" box. S7-GRAPH then signals a supervision error with the output parameter ERR_FLT.
- "Acknowledge errors" is set in the "Sequencer Properties" group. This means that supervision errors occurring during operation must be acknowledged with the input parameter ACK_EF.

Whether or not an error must be acknowledged can be set as "definable" not only in the settings in the dialog box but alternatively also using the relevant parameter (EN_ACKREQ) in the parameter set.

Handling Supervision Errors

Errors that must be acknowledged affect only the relevant sequencer paths, so that the remaining sequencer can be processed while the sequencer path affected is only processed again after the error is acknowledged.


Supervision error occurs
(event V1 enters state)
Actions with V1 are executed once in the next sequencer cycle.
Step activation time Si.U is stopped
Message in FB parameter ERR_FLT
Compilation setting: Acknowledge
Yes
No
Ackn. signal or SV not true?
No
Yes
Ackn. signal?
No
Yes
Si.U:=0
Ackn. signal?
No
Yes
Si.U:=0
Step enabling condition true?
Yes
Yes
Step enabling condition true?
No
No
SV not true?
No
Supervision error leaves state Event V0 enters
Sequencer switches
SV not true?
No
Yes
Supervision error leaves state Event V0 enters
Yes
Supervision error leaves state Event V0 enters
End

7.12 Comment, Extended Name, Number, Name

Step Comment

A step comment can contain a maximum of 2048 characters. It has no influence on the execution of the program.

Step comments are displayed only at the single step display level. If you do not enter an extended step name, the step name is displayed instead.

The Extended Name

The extended name is displayed in the text line above the actions. It can be used, on the one hand, as a simple comment, and on the other hand the extended name can also be used instead of the step name to identify the step to the diagnostic system.

The number of characters in the extended name is not limited. Make sure, however, that you assign extended names that are unique in the program.

If you do not enter an extended name, the step name is displayed instead.

Step Number (for example S1) and Step Name (for example Step1)

The step name and step number must be unique and can be adapted at any of the display levels.

- The step number is automatically assigned by the system (range: 1 to 999).
  If the step number of the first step is set manually in a branch path, when further step-transition pairs are inserted, the numbers are assigned automatically by the system in ascending order starting at the manually entered number. This simply leads to interrupted numbering of the step-transition pairs. You can renumber the steps and transitions at any time so that individual areas are consecutively numbered.
- The step name is automatically assigned by the system (range: step1 to step999).
  It can consist of a maximum of 24 alphanumeric characters. The first character must be a letter. If the zoom factor is too low, the display of step names is suppressed.
  If you modify the name of a step in the "Step Properties" dialog box, S7-GRAPH automatically searches for the old name and replaces it throughout the entire program when you save or compile. All instructions in which the step is referenced are therefore updated.

Transition Number (for example T1) and Transition Name (for example Trans1)

The transition name and transition number must be unique and can be adapted later in all three programming language representations.

- The transition number is assigned automatically by the system (range: 1 to 999).
  If the transition number of the first transition is assigned manually in a branch path, when you insert further step-transition pairs, the numbers are assigned by the system automatically starting at the number entered manually. This simply leads to interrupted numbering of the step-transition pairs. You can renumber the steps and transitions at any time so that individual areas are consecutively numbered.
- The transition name is automatically assigned by the system (range: Trans1 to Trans999).
  It can consist of a maximum of 24 alphanumeric characters. The first character must be a letter. If the zoom factor is too low, the display of the transition name is suppressed. You can modify the transition name in the "Transition Properties" dialog box.

7.13 Programming with Symbolic Addresses

To make the program and sequential control system clearer, you can use addresses with symbolic identifiers or names instead of absolute addresses.

Symbolic addresses are assigned in the symbol table.

The symbol table is automatically created in the S7 program and can be displayed using the menu command **Options > Symbol Table** or in the SIMATIC manager using the "Symbols" folder.

The symbols used are displayed in the symbols folder in the "Variables" tab of the overview window. The corresponding addresses, data types, and comments are displayed in the "Variables" tab of the details window.

Requirements for Using Symbolic Addresses

Symbolic addresses can only be used when a symbolic name has been assigned to the absolute addresses in the symbol table.

Displaying Symbol Information when Entering Addresses

You can simplify the entry of symbolic addresses in conditions and actions by displaying existing symbols in a drop-down list box. If the menu command **View > Display with > Symbol List** is selected, the list box with the symbols is displayed automatically when you enter addresses. If the menu command is not selected, you can open the list box with the symbols at any time using the right mouse button and the **Insert Symbol** command.

Example of Using Symbolic Names

- Instead of the address I0.1, use the defined name “Dr_mot_stopped" or

- Instead of the absolute block identifier FB11, use the symbolic name “Seq_drill".

7.14 Block Parameters and Variables in the S7-GRAPH FB

Each on of the S7-GRAPH FB is assigned a fixed block parameter set. The system provides a selection of three default parameter sets. You can select a parameter set via menu command **Options > Block settings**.

The user can also define block parameters as well as static or temporary variables required for programming.

A variable view is available for editing block parameters and variables. This is found in the "Variables" tab of the overview window. If this window is hidden, you can open it on your desktop via menu command **View > Overviews**.

Further information on these variables are found in the variables detail view of the "Details" window. There you will find information on addresses, initial values and comments referring to the variables.

Inserting Parameters or Variables in the Sequencer

If you want to use a parameter or variable in the sequencer, you can drag the parameter or variable from the "Variables" tab to the sequencer using the mouse and position it at the required location.

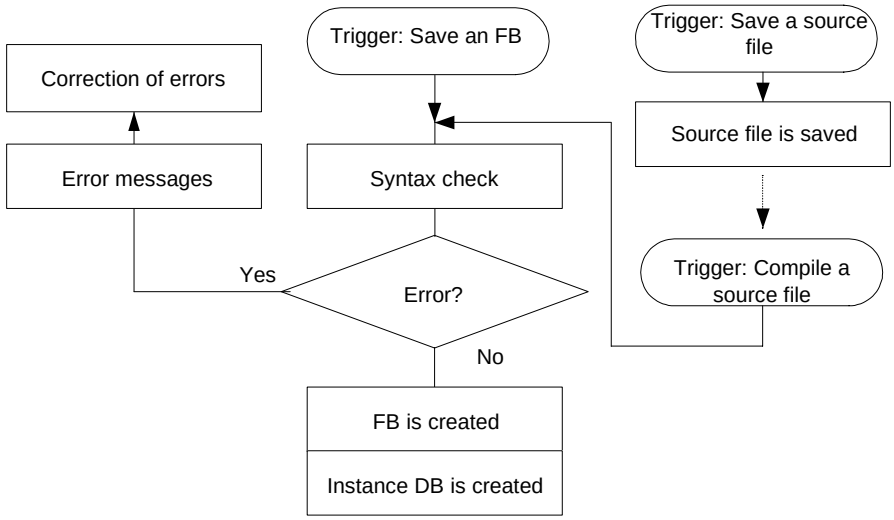
Automatic Update of Changes

Changes you have made to the parameters or variables in the "Variables" tab are automatically updated in the sequencer to exclude any inconsistencies.

8 Saving and Compiling

Principle: Saving and Compiling

After you have programmed a sequencer, you must save the program. When saving and compiling, S7-GRAPH makes a distinction between an S7-GRAPH FB and an S7-GRAPH source file.



8.1 Saving and Compiling

Make sure that you keep to the following rules:

- The size of an S7-GRAPH FB with sequencers is restricted for all
  - CPU 3xx to a maximum of 8 Kbytes or 16 Kbytes.
  - CPU 4xx to 64 Kbytes.
- Each FB has an instance DB that can be understood as the memory of the FB. This contains all the data and parameters required to execute the FB. The instance DB can be created either automatically during compilation or later using a menu command.
- The size of the instance DB depends on the number of programmed steps and transitions but is a minimum of 300 bytes.
- Before saving the S7-GRAPH FB or S7-GRAPH source file, you can start a consistency check.
  - Errors and warnings during the compilation of an S7-GRAPH source file can but do not need to be corrected.
  - Errors during the compilation of an S7-GRAPH FB must, however, be corrected before the FB can be saved.
- If you want to download several sequencers to the PLC at the same time, it is often better to include the system functionality in one of the supplied standard blocks. This reduces the size of every S7-GRAPH block by approximately 5000 bytes. The requirement of the standard block must, however, be added to this (once) (FC70 approx. 8200 bytes, FC71 approx. 7750 bytes, FC72 approx. 10700 bytes, FC73 approx. 8150 bytes).

8.2 Rules for Saving an S7-GRAPH FB

- Only error-free S7-GRAPH FBs can be saved and transferred to the CPU, in other words errors must be eliminated before the FB can be saved. Less significant errors that cause warnings do not need to be eliminated, in other words it is possible to “Save" or “Save As" FBs with warnings.
- When you save an FB, S7-GRAPH implicitly compiles it, in other words the sequencer you have created is checked for syntax errors. Following this, the S7-GRAPH FB does not need to be compiled.
  A detailed window in the "Compile/Decompile Messages" tab displays errors that have occurred (for example jump destination not defined) and/or warnings (for example action without content) as compilation messages.
- A sequencer still containing errors in the S7-GRAPH FB can also be saved as an S7-GRAPH source file with which you can continue to work (menu command **File > Generate Source file**). This allows you to eliminate the errors at a later point in time.
- The block created after compilation is saved in the Blocks folder of the same S7 program.

8.3 Rules for Saving an S7-GRAPH Source File

- Incomplete and syntactically incorrect sequencers can be saved.
- Once an error-free sequencer has been programmed in the S7-GRAPH source file, it can be converted to an S7-GRAPH FB using the "Compile" function.
- Source files are saved in the Source Files folder in the same S7 program.
- Addresses can be saved in source files either in absolute or symbolic format.

Creating a Source File

Follow the steps outlined below:

1. Select the menu command **File > Create Source File**.
2. In the "New" dialog box, select the project and the S7 program and enter the name under which you want to save the source file.

Note

If your S7-GRAPH source file is error-free, you can then create an S7-GRAPH FB with **File > Compile.**

8.4 Settings for Saving and Compiling

Before you save and compile a block, check and correct the settings.

- Select the menu command **Options > Block Settings**. The dialog has three tabs that are relevant for saving and compiling.
  - The "Compile / Save" tab defines the structure and content of the FB and the corresponding DB, some properties of the sequencers and the display of warnings during compilation.
  - The "Messages" tab decides the options for configuring messages.
  - The "Process Diagnostics" tab decides the blocks for which process diagnostics is used.
- Select the menu command **Options > Application Settings**. The dialog has a further tab that is relevant for saving and compiling.
  - The "General" tab determines, among other things, the data to be created when you save your program.

8.4.1 Settings in the "General" Tab

All the settings that you make in this tab are used as the defaults for new blocks.

New Window View

- Here you set the display level at which S7-GRAPH will open a block: Sequencer, single step or permanent instructions.
- Here you select the components to be displayed after a block is opened

Component	Meaning
Comments	A field in which you can enter the block comment or (at the single step level) the step comment.
Symbols	Displays the symbolic names of the addresses in the working window.
Conditions and Actions	A table that displays the assignment of the symbolic names to the absolute addresses at the single step level.
Reference data	Displays the reference data for your program.

On Saving

The settings decide which data are created when you save your program.

- Include instance DB:
  If you select this option, whenever a function block is compiled or downloaded, S7-GRAPH checks whether the instance DB requires updating and if necessary updates it automatically. If you do not select this option, you must create or update the instance DB manually with the menu command **File > Generate Instance DB**.
  Caution
  If you have set the option "Message with ALARM_SQ/ALARM_S (SFC17/SFC18)" in the "Messages" tab (**Options > Application Settings** menu command) the instance DB is automatically included. If the instance DB is deleted, make sure you only recreate it with S7-GRAPH, otherwise messages are lost.
- Generate reference data
  If you select this option, the reference data are created automatically when you save the block. If you do not select the option, you can nevertheless create or update the reference data manually with the menu command **Options > Reference Data**.
- Generate PDIAG data
  If you select this option, S7-GRAPH generates diagnostic data that can be displayed on an operator interface system.
- Use extended names for display system
  Click this check box if you want to use the extended names of the step names for the diagnostic systems.

Addresses in Source Files

The addresses can be saved in source files either in absolute or symbolic format.

- Absolute
  Select this option to access the addresses in absolute format.
- Symbolic
  Select this option to access the addresses in symbolic format. The symbols used must be defined in the symbol table.

Automatically create a new window when a step is opened

- If you select this option, when you double-click a step it is displayed at the single-step level in a separate window.

Save window arrangement on closing

- If you select this option, the settings are saved when you exit the editor and will be displayed again next time you start an editor session.

Maximum number of status jobs used

- You can specify the maximum number of status jobs that will be used by S7-GRAPH.

  All the settings that you make in this tab selected with the menu command **Options > Application Settings** apply to the currently open block.

IEC-compliant

S7-GRAPH as of V5.1 complies with the PLCopen Basis Level for sequential control systems as stipulated in the DIN EN 61131-3 standard.

If you have a sequential control system that conforms with the standard, you can now import it into the STEP 7 data management as an ASCII file using the SIMATIC Manager and then edit it in S7-GRAPH.

To create sequential control systems conforming with the standard, select the "IEC-compliant" option. This setting has the following effects:

- All sequential control systems must contain an initial step. Otherwise, an error is indicated during compilation and no block is created.
- Direct identifiers in source files and blocks (for example M0.0) are preceded by the % character (%M0.0)
- Addresses in source files are always represented by their absolute addresses.
- Compare instructions are displayed without a data type specification if the data type of the address is unambiguous.
- The "Addresses in Source Files Absolute/Symbolic" option is not available and is set to absolute.

To ensure that the source files comply with the PLCopen Standard, the following modifications are made automatically:

- "function_block" is replaced by "program"
- The semicolon is used as the separator
- The "Condition:=" expression is suppressed
- The program name is displayed symbolically in the source file
- The pseudo comment "$_Jump" is no longer required to recognize jumps.

8.4.2 Settings in the "Compile/Save" Tab: FB Parameters

This setting decides the available parameters and the intended functionality of the FB that will be created.

- Minimum: The FB contains only the start parameter INIT_SQ which can be supplied with a value. Only the automatic mode is possible.
- Standard: The FB contains the standard parameters that can be supplied with values. All modes are possible.
- Maximum: The FB contains all the parameters that can be supplied with values. All modes are possible.
- Definable: With this option, you can define your own parameter set in the "Variables" tab in the overview or details window.

Caution

If you edit a user-defined parameter set and then select a standard parameter set later in this tab, you will lose the self-defined parameters.

8.4.3 Settings in the "Compile/Save" Tab: Executability

S7-GRAPH has two options available for creating FBs.

- Full code:
  The entire code required to execute each S7-GRAPH FB is included in the FB. If you have several S7-GRAPH FBs, this means a considerable increase in memory requirements.
- Standard FC required:
  To reduce memory requirements, S7-GRAPH has the two following alternatives: You use a standard FC containing the main code sections for all FBs. This FC is copied to your project automatically when you select this option. The FBs produced using this method are considerably smaller.

  To optimize your memory requirements, it is advisable to use the second method with the standard FC if you have two or more S7-GRAPH FBs.

  You have the choice between the following standard FCs:

  - FC70/FC71
    These two FCs are less than 8 Kbytes in size and can therefore be loaded on a smaller CPU.
    Remember that FC70 and FC71 only support the functionality of S7-GRAPH V4.0. You cannot, for example, program user-defined parameters for the FB, however you can use arithmetic functions, timers, or counters in your program from S7-GRAPH V5.1 onwards.
    FC70 uses the diagnostic functionality of SFC17/18 and can only be used on CPUs that have these functions available. If this is not the case with your CPU, you must use FC71 and do without the diagnostic capability or select "Full Code".
  - FC72
    FC72 provides the entire system functionality but can only be loaded on larger CPUs since it is 11 Kbytes in size.
  - FC73
    FC73 requires less than 8 Kbytes of memory. It can therefore run on all CPUs equipped with sufficient work memory resources for the S7-GRAPH sequencer (+ the remaining user program). If you use this FC, you can considerably reduce the memory requirements of the S7-GRAPH FBs in the same dialog by selecting the "Interface Description: Memory minimized" in the block settings.
    FC73 provides the functionality of S7-GRAPH V5.x. However, the following restrictions apply: The blocks created are not capable of diagnostics and only include status displays when you monitor the sequential control system if you click on the area of the permanent instructions or on the currently active transition.

8.4.4 Settings in the "Compile/Save" Tab: Interface Description

The interface description decides how data for steps and transitions are created in the instance DB. The data can be stored as arrays (ARRAY) or structures (STRUCT) with more or less detailed information.

- Memory minimized
  The interface descriptions are stored in the instance DB as structures. A separate structure with detailed information is created for each step and each transition of the sequencer.
  This option reduces the memory requirements of your S7-GRAPH FB considerably. The blocks created are, however, not capable of diagnostics and only include status displays when you monitor the sequential control system if you click on the area of the permanent instructions or on the currently active transition.
  If you use this option, you must use the supplied FC75 standard block.
- Structure arrays:
  The step and transition descriptions are stored in the instance DB as arrays. This option reduces the memory requirements of your S7-GRAPH FB. The step names are, however, not stored. Symbolic addressing of other blocks is not possible in this case.
- Individual structures:
  The interface descriptions are stored in the instance DB as structures (STRUCT). These contain extremely detailed information about the steps and transitions.
  The blocks created have all the monitoring and diagnostic functions.
  The interface descriptions also include information about the step names so that you have the option of internal and external access using symbolic names.
  The instance DB can be evaluated both using direct access to the data or using the step and transition names.
  The structures increase the size of the instance DB but do not affect the performance when executing the sequencer.
- Interface Description - Download to PLC
  If you select this option, the interface description is also downloaded to the PLC when you download an FB with S7-GRAPH. This allows you to decompile the block on a programming device on which the suitable project structure is available.

8.4.5 Settings in the "Compile/Save" Tab: Sequencer Properties

Sequencer properties influence the way in which sequencers are operated.

- Criteria analysis data in the DB:
  This option is only required if you do **not** want to run process diagnostics with the SIMATIC standard applications ProTool/ProAgent.
  The data required for criteria analysis are written as additional information to the instance DB. They contain a compact form of the condition descriptions of the sequencer.
- Skip Steps
  If both the transition before a step and the transition after the step are valid at the same time, the step does not become active and is skipped.
- Acknowledge Errors
  If a supervision error occurs during operation (conditions of a supervision satisfied), this must be acknowledged with the input parameter "ACK_EF" to allow the controller to continue operation. This option is not available if you have selected the options "FB Parameters: Minimum".
- Synchronization (>= V4.x):
  Synchronization is a function of S7-GRAPH that finds possible synchronization points between the program and process when the sequencer is in the manual mode.

  In the manual mode, in particular, it is often useful to have all the interlocks executed permanently. Here, you can therefore select the permanent execution of interlocks. During operation, the status display (menu command **Debug > Monitor**) displays a missing interlock and with it the step that may have a disturbance.
- Lock operating mode selection:
  If you select this parameter, it is no longer possible to change the mode at a programming device or operator panel (for example in the "Control Sequencer" dialog box).
- Safe activation mode:
  When a step is activated, the system locates and deactivates all steps that cannot be active at the same time as the step being activated due to the sequencer structure.
  If, for example, a step is to be activated in an alternative branch, all steps in the other branch paths will be deactivated.

8.4.6 Settings in the "Compile/Save" Tab: Warnings

During compilation, S7-GRAPH displays error messages and warnings in the "Compile/Decompile Messages" tab. The display of the warnings can be suppressed:

- None: Warnings occurring during compilation are not displayed.
- All: All warnings occurring during compilation are displayed in the "Feedback/Compiler messages" tab

8.4.7 Settings in the "Messages" Tab

Validity of the Settings

- If you click this tab with the **Options > Application Settings** menu command, the settings apply to all newly created blocks.
- If you click this tab with the **Options > Block Settings** menu command, the settings apply to the currently open block.

Options in the "Messages" tab:

Message Handling

- None:
  Messages are not generated.
- Messages with WR_USMSG (SFC52):
  Here, you decide whether messages about interlock or supervision errors are entered in the diagnostic buffer.
  And then send:
  Click this check box, if you also want the messages sent to network nodes (for example OPs).
- Message with ALARM_SQ/ALARM_S (SFC17/SFC18):
  With this option, message numbers and step numbers are entered in the message memory of the CPU and sent from there to the registered nodes.
  To allow an operator panel to display the message text and possibly other information if a disturbance occurs, you must first create an OP project. The data for this are created by S7-GRAPH if you have selected the "Generate PDIAG Data" option in the "General" Tab (menu command **Options > Application Settings**).
  - Interlock with acknowledgment
    If you click this check box, messages about interlock errors must be acknowledged.
  - Supervision with acknowledgment
    If you click this check box, messages about supervision errors must be acknowledged.

The following entries are possible only in the "Application Settings" dialog box. They do, however, also apply to the current block.

Create the Messages:

- Interlock:
  An interlock message can be created as an event message or as an alarm message. Write a message text in the text field. This message text is entered in the OP project along with the diagnostic data. The text is extended dynamically by S7-GRAPH with current information.
- Supervision conditions:
  A supervision message can be created as an event message or alarm message. Write a message text in the text field. This message text is entered in the OP project along with the diagnostic data. The text is extended dynamically by S7-GRAPH with current information.

8.5 Saving

Calculating Memory Requirements

Memory Requirements of a Full-Code FB

With the following formula, you can estimate how much space is required for an S7-GRAPH FB and instance DB in the work memory (n = number of steps).

```
FB = 4900 Byte + n * 130 Byte
DB = 270 Byte + n * 70 Byte
```

The following conditions apply:

- The "with criteria analysis data" option is not selected in the "Compile" tab.
- Each step contains an average of 1.2 transitions, 1.5 actions and 0.8 time monitoring functions.
- Each transition contains an average of 3.5 conditions.

  If many more conditions were programmed per transition, interlock or supervision, the memory requirements can be considerably increased. Per condition, 10-12 bytes and per action 6-8 bytes are required. Complex access, actions with assignments or similar can also greatly increase the memory requirements.

  From this, it is clear that the maximum number of 250 steps is only theoretically possible.

Memory Requirements when Using FC70, FC71, FC72 or FC73

To save space, you can move the system functionality out to one of the supplied standard blocks. This reduces the size of every S7-GRAPH block by approximately 5000 bytes. The requirement of the standard block must, however, be added to this (once) (FC70 approx. 8200 bytes, FC71 approx. 7750 bytes, FC72 approx. 10700 bytes, FC73 approx. 8150 bytes).

You can check which standard FC is suitable for your configuration in the table in the section "Using the Standard Function Blocks FC70, FC71, FC72 and FC73".

Memory requirements when using FC73

If you use FC73, you can considerably reduce the memory requirements of the S7-GRAPH FB. The "Full-Code" option is not available for this minimized memory model.

The following restrictions also apply:

- The blocks created do not have diagnostic capability.
- When monitoring the sequential control system, you do not receive a status display for all processed conditions, but only for the selected condition.

With the following formula, you can estimate how much space is required for an S7-GRAPH FB and instance DB in the work memory (n = number of steps).

FB = 150 Byte + n * 16 Byte
DB = 236 Byte + n * 26 Byte

Each programmed condition requires an additional 2-4 bytes, each action requires 6-8 bytes.

Using Standard Function Blocks FC70, FC71, FC72, and FC73

The following table shows you which FC is suitable for your purposes. If your program is made up of several S7-GRAPH FB, it is advisable to use the same standard function for all FBs so that you do not need to load more than one standard FC on your CPU.

	Full code	**Standard FC70**	**Standard FC71**	**Standard FC72**	**Standard FC73**
Memory requirements	Dependent on the number and size of the sequencers	Approx. 8200 bytes	Approx. 7750 bytes	Approx. 10700 bytes	Approx. 8150 bytes
Restric-tions in terms of the task	No restriction	arithm. functions, timers, and counters can only be used with S7-GRAPH V5.1 and higher	No user-defined parameters, arithm. functions, timers, and counters can only be used with S7-GRAPH V5.1 and higher, no messages with SFC17/SFC18 however messages can be sent with SFC52	No restriction	no messages can be sent and therefore no connection to process diagnostics no step activation time (.U) "Skip Steps" option not possible No user-specific step and transition numbering possible
Restric-tions in terms of the CPU	less than 8 Kbytes and can be loaded on smaller CPUs. particularly suitable for small applications (for example 30 steps)	Can only be loaded on CPUs that have SFC17 and SFC18 (ALARM_S and ALARM_SQ).	No restriction	Due to the 8 Kbyte limit, cannot be loaded on the following: all CPU 313 all CPU314 all CPU314 IFM CPU315-1 (up to version -1AF02)	no restrictions
CPU	no special requirements The 8 Kbyte limit must be adhered to both for the FB and for the DB.	Can only be loaded on CPUs that have SFC17 and SFC18 (ALARM_S and ALARM_SQ). (all S7-400 CPUs and all newer S7-300 CPUs)	No restriction	The CPU must be capable of loading blocks larger than 8 Kbytes (all S7-400 and S7-300 CPUs from Version -1AF03 / -2AF03 of the CPU315 onwards)	No restriction

Saving an S7-GRAPH FB or an S7-GRAPH Source File

The following options are available when you save an open object (block or source file):

1. Select the menu command **File > Save** or click the "Save" button in the toolbar. The object is saved under its present name.
2. If you require a copy of the current object, select the menu command **File > Save As**. The "Save As" dialog appears which you can specify the object name and path of the copy.
3. If you select the menu command **File > Close**, the Close button or the menu command **File > Exit** and the modified object has not yet been saved, you will be prompted to decide whether or not you want to save or discard the modifications or cancel the **Close** command.

Caution

If the sequencer still contains errors, it cannot be saved as an FB. As an alternative, you can save it as an S7-GRAPH source file in which you can continue to work. This allows you to eliminate the errors at a later point in time.

8.6 Compiling

Compiling produces a program that can be understood by the CPU. During compilation, the syntax of the program is checked, the S7-GRAPH FB is created and, if required, the instance DB.

Requirements

Before an FB can be compiled, all the errors must be eliminated. You can have S7-GRAPH check your blocks as follows:

- Either run a consistency check prior to compilation or
- Compile more than once.

Methods for Compiling Sequencers

- **Running the consistency check**
  By running the consistency check, you can detect syntax errors in your program without compiling it. You can use the consistency check both with an S7-GRAPH FB and with an S7-GRAPH source file.
- **Automatic compilation on saving an S7-GRAPH FB**
  If you program the sequencers in an S7-GRAPH FB, compiling is part of saving.
- **Compiling an S7-GRAPH source file**
  If you program the sequencers in an S7-GRAPH source file, you must start compilation explicitly.
- **Eliminating errors after compilation or after the consistency check**
  All errors that occurred during compilation are displayed in a message window following compilation. The message window is anchored to the work area and can be toggled on and off with a button in the toolbar.

Running a Consistency Check

Using a consistency check, you can find syntax errors in your program without compiling it. You can use the consistency check both with an S7-GRAPH FB and with an S7-GRAPH source file.

1. Select the menu command **File > Consistency Check**.
   Result: On completion of the consistency check, the number of errors found is displayed in the "Compile/Decompile Messages" tab and the errors are listed individually.
2. Correct the displayed errors.

Automatically Compiling on Saving an S7-GRAPH FB

If you program the sequencers in an S7-GRAPH FB, the FB is compiled when you save.

1. Select the menu command **File > Save** or click the corresponding button.
   The S7-GRAPH FB is automatically compiled and then saved.
2. Check the information in the "Compile/Decompile Messages" tab for errors and correct them. Save the FB again.

Caution

If errors are indicated, S7-GRAPH has not created an FB. You must eliminate the errors before the S7-GRAPH FB can be saved.

Compiling an S7-GRAPH Source File

If you program your sequencers in an S7-GRAPH source file, you must start the compilation explicitly.

1. Select the menu command **File > Compile** or click the corresponding button.

Result: An S7-GRAPH FB is created automatically from the source file.

2. Check the information in the "Compile/Decompile Messages" tab for errors and correct them. Then recompile your source file.

Caution

If errors are indicated, S7-GRAPH has not created an FB. You must first eliminate the errors before you can compile the file into an FB.

Debugging after Compiling or after the Consistency Check

All the errors that occurred during compilation are displayed in the "Compile/Decompile" messages tab following compilation. This tab is part of the detailed window and can be displayed or hidden with the **View > Details** menu command.
To eliminate the errors, follow the procedure below:

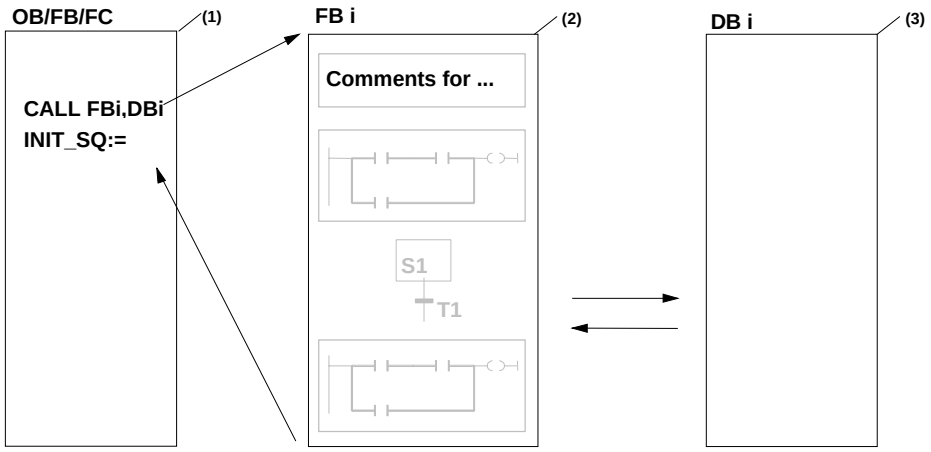
1. Locate the error in the program by double-clicking the error message in the message window.
2. Press the F1 key to obtain a description of the error and how to eliminate it.
3. Correct the sequencer or program as described.

Other errors can be found with the menu commands **Edit > Go To > Previous Error** or **Edit > Go To > Next Error**.

9 S7-GRAPH FB Parameter Assignment and Call

9.1 Calling the S7-GRAPH FB in the S7 Program

Before a loaded S7-GRAPH FB can be executed on the CPU, it must be called in a block that is executed cyclically. This call can be in OB1 or in any other FB or FC called in OB1.



9.2 Calling the S7-GRAPH FB

Program the logic block (for example OB1, FBn, ...), in which the S7-GRAPH FB will be called with one of the STEP 7 programming languages (for example LAD, FBD, STL, SCL):

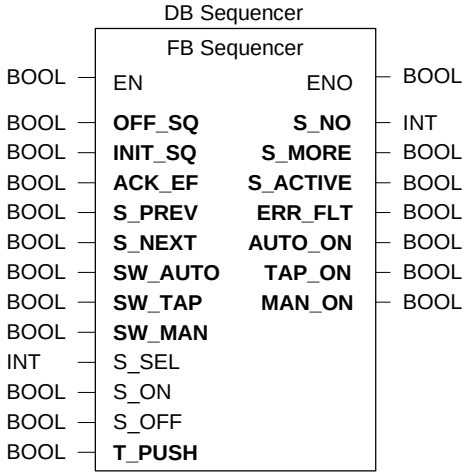
- If you program in Statement List, use the CALL instruction.
- If you program in LAD/FBD, double-click the FB in the program elements catalog.

Note

You only need to assign values to the parameters of the FB that you actually require during operation.

Example: Assignment of the FB for the inching mode

To call and operate the sequencer in the inching mode, you do not require all addresses. The following diagram shows the addresses that must be assigned when programming Ladder Logic. The FB was created with the standard parameter set.



9.3 Selecting an FB Parameter Set

The parameter set you select depends both on the intended use of the sequencer and on the available memory in the CPU. The size of the parameter set also determines the memory requirements for the S7-GRAPH FB and the corresponding instance DB.

Your task ...	**... then use the**		**parameter set**
You want to use your sequencer only in the automatic mode. You do not require any further control or monitoring functions.		→	Minimum
You want to operate your sequencer in various modes. You also require feedback about the process and have the option of acknowledging messages.		→	Standard
Apart from the options provided by the standard parameter set, you require further operator control and monitoring functions for service and commissioning. This parameter set applies to blocks as of version 4.		→	Maximum (as of V4)
Apart from the options provided by the standard parameter set, you require further operator control and monitoring functions for service and commissioning. This parameter set applies to blocks as of version 5x.		→	User-specific/Maximum (as of V5x)

You can also modify existing parameter sets in the variable declaration window. You can, for example, delete unnecessary parameters from a standard parameter set. System parameters cannot be modified however you can define user-specific parameters to meet your requirements.

Caution

Defined parameter names must not be used as step names (and vice-versa).

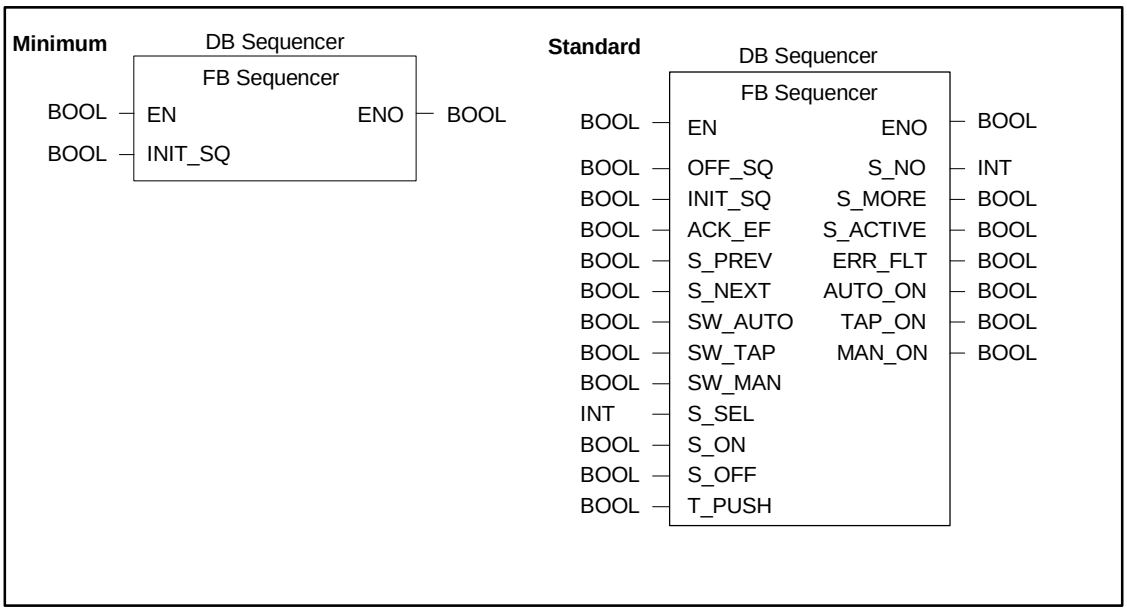
Setting the Parameter Set

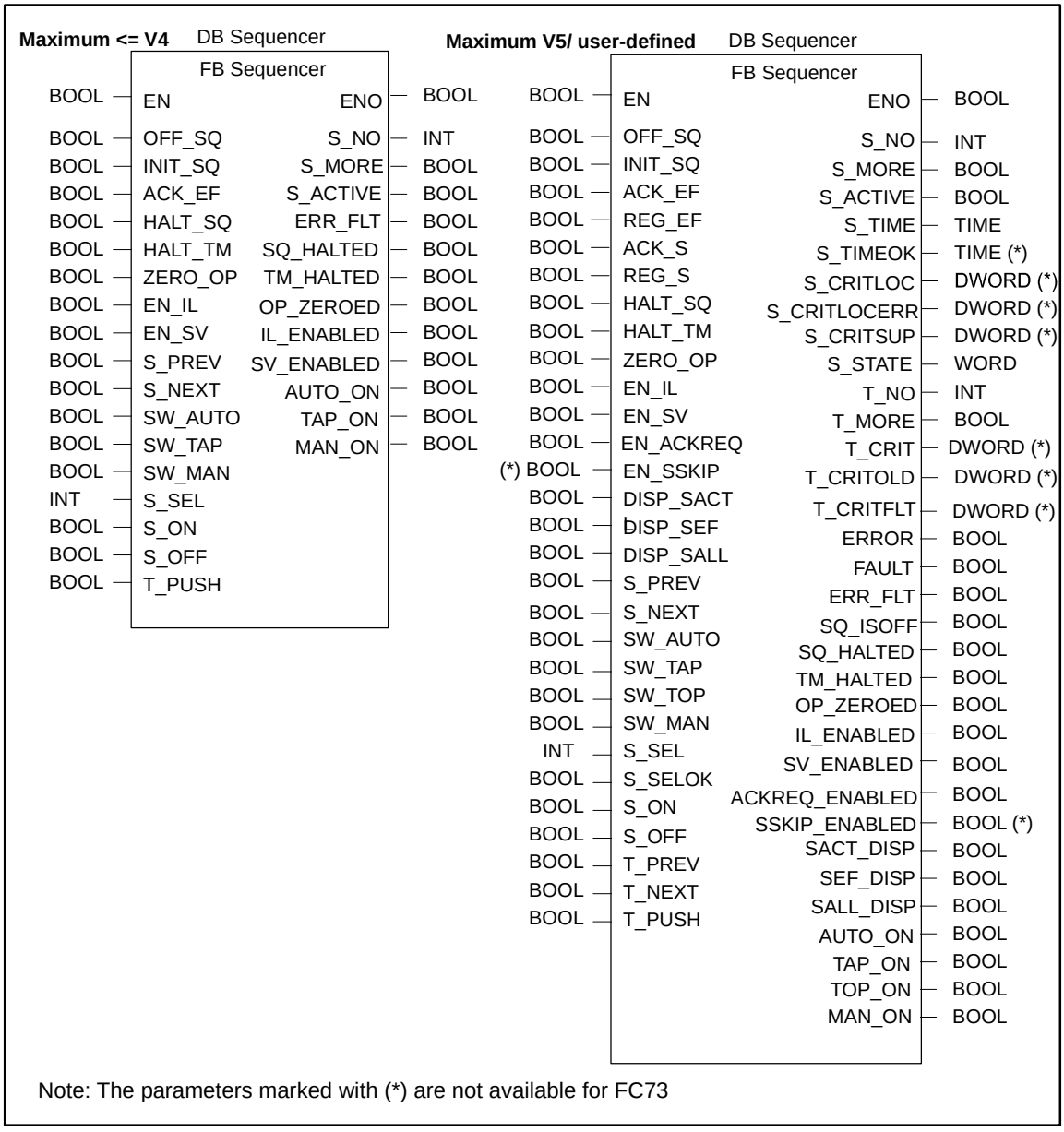
1. Select the menu command **Options > Block Settings** and click the "Compile / Save" tab.
2. Select a setting for the "FB Parameters" entry.

Caution

If you edit a user-defined parameter set and then select a standard parameter set later in this tab, you will lose the self-defined parameters.

9.4 Parameter Sets of the FB





Note: The parameters marked with (*) are not available for FC73

9.5 Input Parameters of the S7-GRAPH FB

The FB reacts to the rising edge of the input parameter (exception EN).

Parameter	Data Type	Description	Min.	Std.	Max.	User-def.
EN	BOOL	Controls execution of the FB (enable input). If EN is not connected, the FB is always executed.	✓	✓	✓	✓
OFF_SQ	BOOL	OFF_SEQUENCE: Sequencer off, in other words deactivate all steps		✓	✓	✓
INIT_SQ	BOOL	INIT_SEQUENCE: Activate initial steps (reset sequencer)	✓	✓	✓	✓
ACK_EF	BOOL	ACKNOWLEDGE_ERROR_FAULT: Acknowledge all errors, force switching to next step		✓	✓	✓
REG_EF	BOOL	REGISTRATE_ERROR_FAULT: Register all errors and disturbances				✓
ACK_S	BOOL	ACKNOWLEDGE_STEP: Acknowledge step indicated in S_NO				✓
REG_S	BOOL	REGISTRATE_STEP: Register step indicated in S_NO				✓
HALT_SQ	BOOL	HALT_SEQUENCE: Stop/reactivate sequencer			✓	✓
HALT_TM	BOOL	HALT_TIMES: Stop/reactivate all step activation times and time-dependent instructions (L and D) of the sequencer			✓	✓
ZERO_OP	BOOL	ZERO_OPERANDS: Reset all addresses of the instructions N, D, L in active steps to zero and do not execute CALL instructions in actions/reactivate addresses and CALL instructions.			✓	✓
EN_IL	BOOL	ENABLE_INTERLOCKS: Deactivate/reactivate interlocks (the sequencer behaves as if interlocks were satisfied)			✓	✓
EN_SV	BOOL	ENABLE_SUPERVISIONS: Deactivate/reactivate supervision conditions (the sequencer behaves as if supervision conditions were not satisfied)			✓	✓
EN_ACKREQ	BOOL	ENABLE_ACKNOWLEDGE_REQUIRED: Activate mandatory acknowledgment				✓
DISP_SACT	BOOL	DISPLAY_ACTIVE_STEPS: Display active steps only				✓
DISP_SEF	BOOL	DISPLAY_STEPS_WITH_ERROR_OR_FAULT: Display steps with errors and disturbed steps only				✓
DISP_SALL	BOOL	DISPLAY_ALL_STEPS: Display all steps				✓

Parameter	Data Type	Description	Min.	Std.	Max.	User-def.
S_PREV	BOOL	PREVIOUS_STEP: Automatic mode: Pages back through the currently active steps. The step number is indicated in S_NO. Manual mode: Indicate previous step (next lower number) in S_NO.		✓	✓	✓
S_NEXT	BOOL	NEXT_STEP: Automatic mode: Page forwards through the currently active steps. The step number is indicated in S_NO. Manual mode: Indicates the number of the next step (next higher number) in S_NO.		✓	✓	✓
SW_AUTO	BOOL	SWITCH_MODE_AUTOMATIC: Mode change: Automatic mode:		✓	✓	✓
SW_TAP	BOOL	SWITCH_MODE_TRANSITION_AND_PUSH: Mode change: Inching mode ("semi-automatic")		✓	✓	✓
SW_TOP	BOOL	SWITCH_MODE_TRANSITION_OR_PUSH: Mode change: Automatic or switch to next				✓
SW_MAN	BOOL	SWITCH_MODE_MANUAL: Mode change: Manual mode (automatic execution is not triggered)		✓	✓	✓
S_SEL	INT	STEP_SELECT: Selects a specific step for the output parameter S_NO. Activate/deactivate in the manual mode with S_ON, S_OFF.		✓	✓	✓
S_SELOK	BOOL	STEP_SELECT_OK: Use value in S_SEL for S_NO				✓
S_ON	BOOL	STEP_ON: Manual mode: Activate the displayed step		✓	✓	✓
S_OFF	BOOL	STEP_OFF: Manual mode: Deactivate the displayed step		✓	✓	✓
T_PREV	BOOL	PREVIOUS_TRANSITION: Display previous valid transition in T_NO				✓
T_NEXT	BOOL	NEXT_TRANSITION: Display next valid transition in T_NO				✓

Parameter	Data Type	Description	Min.	Std.	Max.	User-def.
T_PUSH	BOOL	PUSH_TRANSITION: Transition passes control when the condition is satisfied and T_PUSH (edge) Requirement: Inching (SW_TAP) or automatic or step -by-step (SW_TOP) mode If the block is Version V4 (or earlier), the first valid transition switches. If the block is Version V5 and if the input parameter T_NO is specified, the transition whose number is displayed switches. Otherwise, the first valid transition switches.		✓	✓	✓
EN_SSKIP	BOOL	ENABLE_STEP _SKIPPING: Activate skip step				✓
Parameter set: Min. = Minimum; Std. = Standard; Max. = Maximum; User-def. User-defined						

9.6 Output Parameters of the S7-GRAPH FB

Parameter	Data Type	Description	Min.	Std.	Max.	User-def.
ENO	BOOL	Enable output. When the FB is active and no error has occurred, ENO has the value 1, otherwise 0	✓	✓	✓	✓
S_NO	INT	STEP_NUMBER Display step number		✓	✓	✓
S_MORE	BOOL	MORE_STEPS: Other steps exist and can be selected in S_NO		✓	✓	✓
S_ACTIVE	BOOL	STEP_ACTIVE Displayed step is active		✓	✓	✓
S_TIME	TIME	STEP_TIME Step activation time				✓
S_TIMEOK	TIME	STEP_TIME_OK: No error in step activation time				✓
S_CRITLOC	DWORD	STEP_CRITERIA Interlock criteria bits				✓
S_CRITLOC ERR	DWORD	S_CRITERIA_IL_LAST_ERROR: Interlock criteria bits for event L1				✓
S_CRITSUP	DWORD	STEP_CRITERIA Supervision criteria bits				✓
S_STATE	WORD	STEP_STATE: Step state bits				✓
T_NO	INT	TRANSITION_NUMBER: Valid transition number				✓
T_MORE	BOOL	MORE_TRANSITIONS: Other valid transitions available for display				✓
T_CRIT	DWORD	TRANSITION_CRITERIA: Transition criteria bits				✓
T_CRITOLD	DWORD	T_CRITERIA_LAST_CYCLE: Transition criteria bits from last cycle				✓
T_CRITFLT	DWORD	T_CRITERIA_LAST_FAULT: Transition criteria bits for event V1				✓
ERROR	BOOL	INTERLOCK_ERROR: Interlock error (any step)				✓
FAULT	BOOL	SUPERVISION_FAULT: Supervision error (any step)				✓
ERR_FLT	BOOL	IL_ERROR_OR_SV_FAULT: Group disturbance		✓	✓	✓
SQ_ISOFF	BOOL	SEQUENCE_IS_OFF: Sequencer idle (no step active)				✓
SQ_HALTED	BOOL	SEQUENCE_IS_HALTED: Sequencer stopped			✓	✓
TM_HALTED	BOOL	TIMES_ARE_HALTED: Timers stopped			✓	✓
OP_ZEROED	BOOL	OPERANDS_ARE_ZEROED: Addresses reset			✓	✓

Parameter	Data Type	Description	Min.	Std.	Max.	User-def.
IL_ENABLED	BOOL	INTERLOCK_IS_ENABLED: Interlock enabled			✓	✓
SV_ENABLED	BOOL	SUPERVISION_IS_ENABLED: Supervision enabled			✓	✓
ACKREQ_ENABLED	BOOL	ACKNOWLEDGE_REQUIRED_IS_ENABLED: Mandatory acknowledgment activated				✓
SSKIP_ENABLED	BOOL	STEP_SKIPPING_IS_ENABLED: Skip step activated				✓
SACT_DISP	BOOL	ACTIVE_STEPS _WERE_DISPLAYED: Display active steps only in S_NO				✓
SEF_DISP	BOOL	STEPS_WITH_ERROR_FAULT_WERE_DISPLAYED: Display only steps with error and disturbed steps in S_NO				✓
SALL_DISP	BOOL	ALL_STEPS_WERE_DISPLAYED: Display all steps in S_NO				✓
AUTO_ON	BOOL	AUTOMATIC_IS_ON: Indicates the automatic mode		✓	✓	✓
TAP_ON	BOOL	T_AND_PUSH_IS_ON: Indicates the inching mode		✓	✓	✓
TOP_ON	BOOL	T_OR_PUSH_IS_ON: Display SW_TOP mode				✓
MAN_ON	BOOL	MANUAL_IS_ON: Indicates the manual mode		✓	✓	✓
Parameter set: Min. = Minimum; Std. = Standard; Max. = Maximum; User-def. = User-defined						

9.7 The Modes of the Sequential Control System

By selecting one of the four possible modes, you decide how the sequencer progresses from step to step. By setting suitable parameters for the S7-GRAPH FB, you can select the mode of the sequential control system as required. The following modes are available:

- Automatic Mode:
  In the automatic mode control passes to the next step when a transition is satisfied.
- Manual mode:
  In contrast to the automatic mode, in the manual mode, the next step in not enabled when the transition is satisfied. The steps are selected and deselected manually.
- Inching mode:
  The inching mode corresponds to the automatic mode with an additional step enabling condition. Not only must the transition be satisfied, but there must also be a rising edge at the T_PUSH parameter before control passes to the next step.
- Automatic or switch to next
  In the "automatic or step-by-step" mode, control is passed to the next step when the transition is satisfied or when there is a rising edge at the T_PUSH parameter.

Requirements for Operation in Different Modes

Automatic mode is the standard mode for sequencers. It is the default for all parameter sets of the S7-GRAPH FB.

To be able to select a different mode, you must compile the S7-GRAPH FB with the standard, maximum or definable parameter set.

Selecting Modes

You select one of the four modes using the input parameters of the FB. Apply a rising edge to one of the following parameters to activate the corresponding mode.

- SW_AUTO (Automatic Mode)
- SW_MAN (Manual Mode)
- SW_TAP (Inching Mode)
- SW_TOP (Automatic or Switch to next)

9.7.1 Automatic Mode (SW_AUTO)

In the automatic mode control passes to the next step when the transition is satisfied.

Input Parameters for the Automatic Mode

Parameters not listed are not mode-dependent.

Parameter	Data	Description
OFF_SQ	BOOL	Sequencer off; in other words, deactivate all steps
INIT_SQ	BOOL	Initialize sequencer, jump to initial step
ACK_EF	BOOL	Acknowledge all errors, force switching to next step
S_PREV	BOOL	Pages back through the currently active steps. The step number is indicated in S_NO.
S_NEXT	BOOL	Page forwards through the currently active steps. The step number is indicated in S_NO.
SW_TAP	BOOL	Request inching mode
SW_MAN	BOOL	Request manual mode
SW_TOP	BOOL	Set automatic or switch to next mode

Output Parameters of the Automatic Mode

Parameter	Data	Description
S_NO	INT	Displays the active step selected with S_PREV or S_NEXT
S_MORE	BOOL	Further active steps exist and can be selected in S_NO
S_ACTIVE	BOOL	The step indicated in S_NO is active
ERR_FLT	BOOL	Error/disturbance occurred
AUTO_ON	BOOL	Automatic mode is active
TAP_ON	BOOL	Inching mode is not active
MAN_ON	BOOL	Manual mode is not active
TOP_ON	BOOL	Automatic or switch to next mode is not activated

9.7.2 Manual mode (SW_MAN)

The manual mode is selected, for example, to test the sequencer. The manual mode is characterized as follows:

1. Next step not enabled when a transition is satisfied
2. Steps can be manually selected and deselected

Input Parameters of the Manual Mode

Parameters not listed are not mode-dependent.

Parameter	Data	Description
OFF_SQ	BOOL	All steps are immediately deactivated, in other words "Sequencer Off"
INIT_SQ	BOOL	Initialize sequencer, jump to initial step
ACK_EF	BOOL	Acknowledge all errors
S_PREV	BOOL	Indicate previous step in S_NO
S_NEXT	BOOL	Indicate next step in S_NO
SW_AUTO	BOOL	Request the automatic mode
SW_TAP	BOOL	Request inching mode
SW_TOP	BOOL	Set automatic or switch to next mode
S_SEL	INT	Step number for step selection, indicated in S_NO following selection
S_ON	BOOL	The step indicated in S_NO is activated
S_OFF	BOOL	The step indicated in S_NO is deactivated

Output Parameters of the Manual Mode

Parameter	Data	Description
S_NO	INT	Step number, indicates the selected step
S_MORE	BOOL	Other steps exist and can be selected in S_NO.
S_ACTIVE	BOOL	The step indicated in S_NO is active
ERR_FLT	BOOL	Error/disturbance occurred
AUTO_ON	BOOL	The automatic mode is not active
TAP_ON	BOOL	Inching mode is not active
MAN_ON	BOOL	The manual mode is active
TOP_ON	BOOL	Automatic or switch to next mode is not activated

Caution

The input parameters DISP_SACT, DISP_SEF, DISP_SALL influence the number of steps displayed at the S_NO parameter.

In the manual mode, an unsuitable parameter assignment can mean that no step is displayed.

If you want to display all the steps, set the DISP_SALL parameter to TRUE or delete the three input parameters DISP_SACT, DISP_SEF, DISP_SALL.

9.7.3 Inching mode (SW_TAP)

To install or test a system step by step, the inching mode can be very useful:

- The sequencer passes control when the transition is satisfied and there is an edge change from 0 to 1 at the parameter T_PUSH.

Input Parameters of the Inching Mode

Parameters not listed are not mode-dependent.

Parameter	Data	Description
OFF_SQ	BOOL	All steps are immediately deactivated, in other words "Sequencer Off"
INIT_SQ	BOOL	Initialize sequencer, jump to initial step
ACK_EF	BOOL	Acknowledge all errors
S_PREV	BOOL	As for automatic
S_NEXT	BOOL	As for automatic
SW_AUTO	BOOL	Request the automatic mode
SW_MAN	BOOL	Request manual mode
SW_TOP	BOOL	Set automatic or switch to next mode
T_PUSH	BOOL	The transition passes control when its conditions are satisfied and there is a rising edge at T_PUSH.

Output Parameters of the Inching Mode

Parameter	Data	Description
S_NO	INT	Step number, indicates the selected step
S_MORE	BOOL	Further active steps exist and can be selected in S_NO
S_ACTIVE	BOOL	The indicated step is active
ERR_FLT	BOOL	Error/disturbance occurred
AUTO_ON	BOOL	The automatic mode is not active
TAP_ON	BOOL	The inching mode is active
MAN_ON	BOOL	Manual mode is not active
TOP_ON	BOOL	Automatic or switch to next mode is not activated

9.7.4 Automatic or Switch to Next Mode (SW_TOP)

For initial project start up or for testing a control system step-by-step, the "automatic or switch to next" mode can be extremely useful:

- The sequencer progresses when the Transition is satisfied or when there is an edge change from 0 to 1 at the parameter T_PUSH.

Input Parameters of the Automatic or **Switch to next** Mode

Parameters not listed are not mode-dependent.

Parameter	Data	Description
OFF_SQ	BOOL	All steps are immediately deactivated, in other words "Sequencer Off"
INIT_SQ	BOOL	Initialize sequencer, jump to initial step
ACK_EF	BOOL	Acknowledge all errors
S_PREV	BOOL	As for automatic
S_NEXT	BOOL	As for automatic
SW_AUTO	BOOL	Request the automatic mode
SW_MAN	BOOL	Request manual mode
SW_TAP	BOOL	Request inching mode
T_PUSH	BOOL	The transition passes control when there is a rising edge at T_PUSH regardless of whether the transition is satisfied.

Output Parameters of the Automatic or Switch to Next Mode

Parameter	Data	Description
S_NO	INT	Step number, indicates the selected step
S_MORE	BOOL	Further active steps exist and can be indicated in S_NO
S_ACTIVE	BOOL	The indicated step is active
ERR_FLT	BOOL	Error/disturbance occurred
AUTO_ON	BOOL	The automatic mode is not active
TAP_ON	BOOL	Inching mode is not active
MAN_ON	BOOL	Manual mode is not active
TOP_ON	BOOL	Automatic or switch to next mode is activated

9.7.5 Selecting a Step with S_SEL

1. Set the manual mode using the input parameter SW_MAN (edge change from 0 to 1).
2. Enter the number of the step you want to select using the input parameter S_SEL. If the S_SELOK parameter exists in the compilation model, you must confirm the adoption of the step number with the parameter. The step is then included in the display.
3. Activate the selected step with the input parameter S_ON (edge change from 0 to 1).

If the sequencer has paths in simultaneous branches and you want to select more than one step, repeat steps 2. and 3.

9.7.6 Selecting a Step with S_PREV or S_NEXT

1. Depending on whether you want to page forwards or backwards,
   - select a previous step (lower step number) with the input parameter S_PREV,
   - select a later step (higher step number) with the input parameter S_NEXT.
2. Activate the selected step with the input parameter S_ON (edge change from 0 to 1).

Note

Remember that this procedure can lead to somewhat longer reaction times.

9.7.7 Progressing to the next Step with T_PUSH

1. Set the inching mode using the input parameter SW_TAP or the automatic or step-by-step mode using the input parameter SW_TOP.
2. Apply an edge to the T_PUSH input parameter so that the transition switches. If you have set the SW_TAP parameter, the sequencer only progresses when the transition is satisfied. If you have set SW_TOP, the sequencer progresses even if the transition is not satisfied.

Caution

- If the block is Version V4 (or earlier), the first valid transition switches.
- If the block is Version V5 and if the input parameter T_NO is specified, the transition whose number is displayed switches. If there is no T_NO input parameter, the first valid transition switches.

---

10 Downloading the S7-GRAPH-FB and Instance-DB

10.1 Downloading Blocks from the Programming Device to the CPU

Requirements for Downloading the S7-GRAPH FB to the CPU

To download the user program to the CPU, the following requirements must be met:

- There is a connection between the programming device and the programmable logic controller.
- The sequential control system consisting of the FB, instance DB and, for example, OB1 has been created.
  - The S7-GRAPH FB was compiled free of errors.
  - The instance DB for the S7-GRAPH FB has been created.
- Whenever possible, download the S7-GRAPH block in the STOP mode since following downloading of the instance DB, the sequencer is automatically set to the initial state.
- The S7-GRAPH FB is called in a block (for example OB1) that is executed cyclically, and the block has already been downloaded to the CPU.

Download with instance DB:

If you select the option "Include Instance DB" in the "General" tab selected with the menu command **Options > Application Settings** , S7-GRAPH checks whether the instance DB has changed since it was last downloaded and whether it must be updated. The option "Download with instance DB" is selected when necessary in the "Download" dialog box. The instance DB must only be downloaded along with the FB to the CPU in this situation.

Otherwise, S7-GRAPH does not check the loaded instance DB and leaves you to decide whether or not to transfer the instance DB again. The option "Download with Instance DB" is not selected in the "Download" dialog box.

Download with Standard FC

If you select this option with the **Options > Block Settings** menu command in the "Compile / Save" tab, the system checks whether the specified standard FC already exists online on the CPU. If the FC does not exist, it is made available for downloading.

Turn off the sequencer before download

If this option is selected, the active structure element of the sequencer, the active transition or the active step is stopped when you download the program to the CPU.

If the instance DB is downloaded as well, the function block starts at the initialization step.

If the instance DB is not downloaded as well, the structural element active before the download is activated again on completion of the download.

Downloading to the CPU

To download the S7-GRAPH FB with its instance DB to the CPU, follow the steps outlined below:

1. With the FB open, click the menu command **PLC > Download**. The FB currently open in the working window is downloaded to the CPU.
2. Make the relevant entries and selections in the "Download" dialog box:
   - If you have clicked "Include instance DB", the system automatically prepares the values according to the actual situation. You can change this default.
   - If you have not clicked "Include instance DB", you yourself must decide whether or not the instance DB is downloaded as well.
   - If the FB requires a standard FC, you can specify whether or not this FC is downloaded as well. If the FC already exists online, this option is deactivated.
3. If the blocks already exist on the CPU, confirm the prompt to overwrite the blocks.

Caution

You should only download S7-GRAPH blocks in the RUN mode when the sequencer is in the initial state or in the OFF state. If you download the blocks the sequencer in a different state, when overwriting an old block, problems may occur in the synchronization of the sequencer with the process. For example, both latching and non-latching actions of the steps active at the time of downloading are not reset.

If it is not possible to change the sequencer to the initial state or to the OFF state, deactivate all steps (for example using the "Control Sequencer" function) before you download the blocks to the CPU.

10.2 Uploading Blocks from the CPU to the Programming Device

Requirements for Uploading Blocks to the Programming Device

To upload an FB from the CPU, there must be a connection between the programming device and the programmable controller.

Uploading from the CPU to the Programming Device

It is possible to upload blocks from the CPU using the SIMATIC Manager of STEP 7. The SIMATIC Manager decompiles the blocks to S7-GRAPH. You can then edit the sequential control system as described in Section 5. For more information about uploading from the CPU to the programming device, refer to the online help of the SIMATIC Manager.

Remember that you can only decompile again to a PG that does not contain the project structure if you click the option "Interface Description - Download to PLC" in the "Compile/Save" tab of the "Block Settings" dialog box.

11 Monitoring and Testing the Sequential Control System

Using the Monitoring and Test Functions

The monitoring functions allow you to monitor and check a program while it is being executed on the CPU.

This allows you to locate errors that were not indicated by the formal consistency check while you were writing the program or by the syntax check performed during compilation. These errors include the following:

- Programming errors, for example incorrectly set monitoring times in the supervision conditions
- Logical errors in the program structure, in other words the program steps and conditions do not match the actual process sequence you require.

Methods of Testing the Sequential Control System

S7-GRAPH provides you with a variety of methods with which you can test a sequential control system. These methods can be divided into the following classes:

- Monitoring the status information of the sequential control system
  With this function, you obtain a fast overview of the way in which the sequencers function.
- Controlling the sequencer
  Apart from controlling the sequencer using the programmed FB call, S7-GRAPH also provides a convenient dialog for the test mode.
- Synchronization
  Synchronization is an S7-GRAPH function with which the sequencer can be matched to the process.
- Extended test functions for monitoring the sequential control system
  By calling STEP 7 functions from within S7-GRAPH, you can obtain further, comprehensive information about the entire process.

Requirements for Activating the Test Functions of S7-GRAPH

Before you can test an S7-GRAPH program, the following requirements must be met:

1. The PG must be connected online to the CPU.
2. The program was compiled without errors.
3. The S7-GRAPH FB must be called in a block that is executed cyclically (for example OB1).
4. The program (in other words FB, DB OB and, if applicable, standard FCs) must be loaded on the CPU.
5. The CPU must be in the RUN (read) mode or RUN-P (read and write) mode.

11.1 Monitoring and Test Functions in the S7-GRAPH

11.1.1 Monitoring the Status Information

Principle of the S7-GRAPH Monitoring Function

When you start the monitoring function, you also start the test mode for checking the sequential control system. The individual steps and conditions and the signal state of the addresses are displayed in different colors on the screen depending on the status. The step-by-step execution of the program can therefore be followed visually on the screen.

Depending on the current state of the sequencer, the status is displayed for the following components:

- Steps and conditions
- Signal state of the addresses

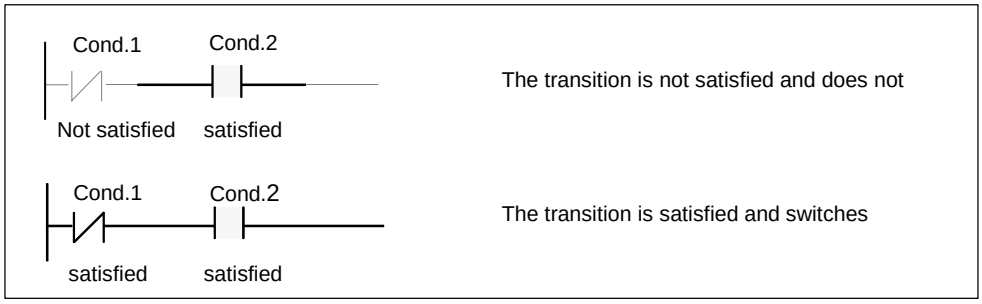
Changing the Color Settings for the Status Display

You can select and set the required color for a particular status display individually using the menu command **Options > Application Settings** in the "Editor" tab of the "Application Settings" dialog box.

Status Display for Steps and Conditions

The status of the following elements of the sequencer is shown in color:

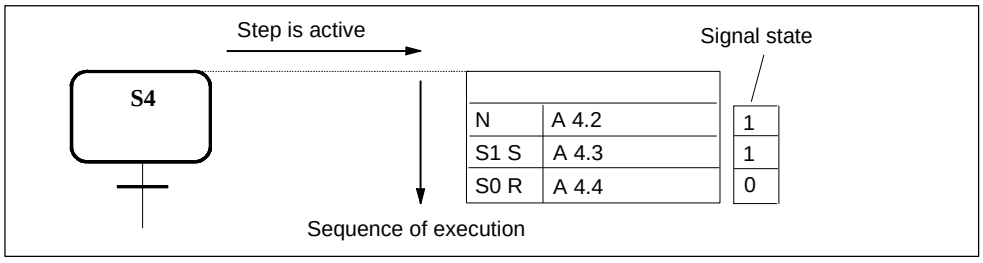
- Steps: Active and disturbed steps are shown in different colors.
- Permanent conditions, transitions, interlocks and supervisions:
  - The display indicates whether individual conditions are satisfied (for example a memory bit or an output).
  - The display indicates whether a complete logic operation consisting of several conditions is satisfied, for example an OR operation.
  - The display indicates whether the entire transition (T), interlock (C) or supervision (V) is satisfied.



Status Display for the Signal State of Addresses

In steps, the signal states of the programmed actions are displayed. The signal state of the addresses (for example inputs, outputs, memory bits) is determined by the actual query of the individual action. This means it is not purely a step-specific evaluation. Boolean addresses are displayed in this status display. The status values of other addresses are displayed in the details window in the "Addresses" tab.

If there are block calls within the actions, there is no status display.



11.1.2 Control Sequencer

Control Sequencer is a test function with which you can test the sequencer with S7-GRAPH in all modes. All the settings and entries for the dialog box have the same effect as the corresponding FB parameters.

The entries in the "Sequencer Control" dialog box can be different from the settings you used to compile the sequencers. The dialog box settings have priority.

"Sequencer Control" Dialog Box

The "Sequencer Control" dialog box is used both as an output field that displays the current settings and at the same time as an input field in which you can change the current status**GlossStatus**.

If you acknowledge an error, initialize the sequencer or want to change a step in the manual mode, call the "Control Sequencer" dialog box with the menu command **Debug > Control Sequencer**.

If you modify these settings in the dialog box, the originally selected settings are indicated in bold face so that you can return to the original setting at any time.

To be able to make modifications, the mode selector on the CPU must be set to the RUN-P position.

11.1.3 Synchronization

S7-GRAPH helps you to locate possible synchronization points between the process and sequencer.

A process is no longer synchronized when it is changed to a different status**GlossStatus** manually. This can, for example, be the result when you change to the manual mode in which you can activate any step even if the previous transition is not satisfied. To be able to continue the process automatically and to locate possible synchronization points, you can start the Synchronization function.

You can choose two different strategies for the synchronization:

S7-GRAPH indicates all steps for which the previous transition is satisfied and the transition following the step is not satisfied.

Requirements

- The sequencer is being controlled in the manual mode.
- You have selected your synchronization strategy in the "Application settings - General" dialog box.
- The block is compiled with the option "Synchronization". You can find this option in the tab "Compile / Save" (menu command **Options > Block settings**).

Starting Synchronization

To start synchronization:

1. Select the menu command **Debug > Synchronization**.
   S7-GRAPH now searches for all steps that meet the synchronization condition. All steps found are marked yellow.
2. Select one or more of these steps to be selected using the mouse pointer that changes to a crosshair. You can also select other steps that are not marked yellow.

Caution

In a simultaneous branch, each path must contain a step to be activated.

---

3. Activate the selected steps with the "Activate" button.
4. Now switch the sequencer back to the automatic mode.

11.2 Test Functions of STEP 7

The following test functions are available for further analysis of the sequential control system. Please note, however, that not all the functions available provide information about an S7-GRAPH sequential control system:

- Monitor and Modify Variables
- Module Status
- Query the Operating Mode
- Evaluate CPU Messages
- Create Reference Data
- Create Process Diagnostics Data
- Check Block Consistency

These functions provide you with further methods of debugging and supply important information about the use of blocks and addresses. You can also select these test functions directly in the SIMATIC Manager.

Note

If you require more detailed information about these topics, display the help texts relating to the menu commands in the SIMATIC Manager.

Monitoring and Modifying Variables

With the menu command **PLC > Monitor/Modify Variables**, you can create a variable table and monitor and modify the signal states of addresses (for example input I2.3).

The signal states to be monitored can be put together individually in the variable table by entering the required addresses in the displayed table.

Querying Module Information and the Operating Mode

With the menu command **PLC > Module Information**, you can obtain a variety of information about modules. As additional information on the sequential control system, the content of the diagnostic buffer and the current utilization of the work and load memory can be particularly interesting.

You can, for example, find out the reasons for errors using the "Diagnostic Buffer" and "Stacks" tabs.

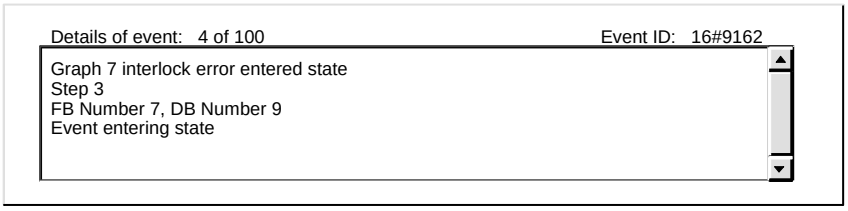
With the menu command **PLC > Operating Mode**, you can query and modify the current mode of your CPU, for example restart.

Evaluating the Diagnostic Buffer

If you have set the option "Message with WR_USMSG (SFC52)" with the menu command **Options > Block Settings** in the "Messages" tab, S7-GRAPH enters the following messages in the diagnostic buffer:

- Interlock errors entering/leaving state
- Monitoring errors (execution errors) entering/leaving state

From the detailed information about the message, you can found out where the error occurred.



Evaluating the CPU Messages

In the SIMATIC Manager, you can use the menu command **PLC > S7 Message** to display event and alarm messages.

Displaying Reference Data

To test the sequential control system, you can call up a variety of reference data. You create the reference data as follows:

- When you save the FB, reference data are created if you have selected this function in the "General" tab which you can display with the menu command **Options > Application Settings**. The "Generate Reference Data" option must be selected.
- With the menu command **Options > Reference Data**, the reference data are created when required. Following this, S7-GRAPH calls the STEP 7 dialog box to display the reference data.

The following table shows the information available:

List	**Content of the list**
Cross-reference	Overview of the use of addresses in the memory areas I, Q, M, P, T, C and DB within the user program.
Program structure	Call hierarchy of the blocks within a user program and overview of the blocks used and their interdependencies.
Reference list	Shows the use of: • Inputs, outputs and memory bits • Timers and counters
List of unused addresses	Overview of all the symbols found in the symbol table but not used in the parts of the user program for which reference data exist.
List of addresses without symbols	Overview of all the absolute addresses that are used in parts of the user program for which reference data exist but for which there is no symbol defined in the symbol table.

If the reference data "Program Structure" or "Cross-Reference List" are displayed, you can jump to the point in the user program at which a selected block or address is used with the menu command **Edit > Go To > Location**.

Generating and Updating Diagnostic Data

Diagnostic data are generated when you save the FB if you select the "Generate PDIAG Data" option in the "General" tab of the "Application Settings" dialog box. You must also select the "Message with ALARM_SQ/ALARM_S" option in the "Messages" tab of the "Block Settings" dialog box.

Updating Diagnostic Data After Rewiring

If you have modified addresses in the SIMATIC manager using the menu command **Rewire**, and the addresses are also used by the sequencer, the corresponding diagnostic data are only updated when you open the corresponding S7-GRAPH FB again and save it.

Check Block Consistency

After you modify a block interface, you must adapt all blocks that call this block otherwise you might cause program inconsistencies and time stamp conflicts.

You can call the STEP 7 "Check Block Consistency" function to initiate a global consistency check of all program changes in S7 blocks. This lets you to gain control over the effects of interface modifications at other blocks and to quickly eliminate errors.

The function will automatically jump to the corresponding positions in the Editor if inconsistencies which could not be resolved automatically. There you can edit all relevant block elements until all inconsistencies are resolved.

Procedure:

Select the menu command **Edit > Check Block Consistency**.

Further information on this function is found in the STEP 7 Help.

12 Printing the Sequencer

12.1 Printing the Sequencer

Documenting the Program

After you have created the sequential control system, you can print it out at any time to create your documentation. The printout is either according to the settings or the content of the active window is printed. The display level and zoom factor set for the active window are taken into account in the printout.

You can modify the page layout to suit your purposes and check it in the print preview. You can adapt the following:

- Display level and zoom factor
- Header and footer of the printed page
- Page format

Appearance of the Printout

If you have a multi-page printout, references to the adjacent pages are printed at the margins of the pages to allow better orientation. References to previous and following elements are also printed if, for example, consecutive steps or steps and transitions are split on two pages due to a form feed.

To improve clarity, the start of a transition can be printed right-aligned and the action block left-aligned. There is adequate clearance between sequencer paths. Pages with no content are not printed.

Requirements for Printing

The S7-GRAPH print function will already be familiar to you from other Windows applications. Before you can print, the printer must be installed and initialized using the Windows Control Panel.

Appearance of the Printout

In the "Print" dialog box, you can select the content of the current printout.

- Current View: The view of the FB as selected in the working area is transferred to the printer.
- According to Settings: Regardless of the current display, the settings you made in the "Print" tab of the "Application Settings" dialog box are used.

"Print" Tab, "Application Settings" Dialog Box

in the "Print" tab of the "Application Settings", you can specify the standard settings for printouts. You select the required settings by placing a check mark. You can include the following components in the printout:

- **Sequencer** view with / without conditions and actions
- **Step** view with / without address List, step comment and cross references
- **Permanent instructions** with / without address list
- **Addresses**
- **Block properties**
- **Compiler options**
- **Variables**

The active block can also be displayed in symbolic / absolute format or as a LAD/FBD block.

Creating the Headers and Footers in the SIMATIC Manager

To specify the header and footer lines for your documents, use the menu command **File > Headers and Footers** in the SIMATIC manager.

Selecting the Page Format

With the menu command **File > Page Setup**, you can select the page format you require for your printout. The format set in the SIMATIC Manager is taken as the default.

Checking the Current Block in the Print Preview

With the menu command **File > Print Preview**, you can check all the settings for the current block before you send the document to the printer. Editing is not possible in this view.

Starting the Print Job

You can start the print function in two ways in S7-GRAPH:

- Select the "Print" button in the toolbar.
- Select the menu command **File > Print**.

A dialog box is then opened in which you can select various print options such as the printer and number of copies.

Confirm with OK to send the document to the printer.

13 Overview of all Actions

S7-GRAPH uses the settings for the mnemonics (German or English) selected in the "Language" tab in the SIMATIC manager with the menu command **Options > Customize**.

13.1 Standard Actions with and without Interlock

These actions are executed each time the sequencer is run through.

Action					**Explanation**	**Address range**
Event	**Instruction**	**Address**	**Location**	**Time Constant**		
	N	Q, I, M, D∗	m.n		When Schritt is active, the signal state of the address = 1.	0.0 to 65535.7
	S	Q, I, M, D∗	m.n		When the step is active, the address is set to 1 and then remains set to 1.	0.0 to 65535.7
	R	Q, I, M, D∗	m.n		When the step is active, the address is set to 0 and then remains set to 0.	0.0 to 65535.7
	D	Q, I, M, D∗	m.n	T#<const>	n seconds after step activation the signal state of the address is 1 for the duration of the step activation. This does not apply if the step is active for a time shorter than n seconds.	0.0 to 65535.7
	L	Q, I, M, D∗	m.n	T#<const>	If the step is active, the address has signal state 1 for n seconds.	0.0 to 65535.7
	CALL	FB, FC, SFB, SFC	Block number		As long as the step is active, the specified block is called.	
	NC	Q, I, M, D∗	m.n		As long as the step is active and the condition (interlock) is satisfied, the signal state of the address is 1.	0.0 to 65535.7
	SC	Q, I, M, D∗	m.n		As long as the step is active and the condition (interlock) is satisfied, the address is set to 1 and then remains set to 1.	0.0 to 65535.7
	RC	Q, I, M, D∗	m.n		As long as the step is active and the condition (interlock) is satisfied, the address is set to 0 and then remains set to 0.	0.0 to 65535.7
	DC	Q, I, M, D∗	m.n	T#<const>	n seconds after step activation and as long as the step is active and the condition (interlock) is satisfied, the signal state of the address is 1. If the step is not active, the signal state of the address is 0.	0.0 to 65535.7

Action					Explanation	Address range
Event	Instruction	Address	Location	Time Constant		
	LC	Q, I, M, D*	m.n	T#<const>	If the step is active and the condition (supervision) is satisfied, the address has the signal 1 for n seconds. If the step is not active, the address has signal 0.	0.0 to 65535.7
	CALLC	FB, FC, SFB, SFC	Block number		As long as the step is active and the condition (interlock) is satisfied, the specified block is called.	

An instance DB is required with CALL[C] FB/SFB.

13.2 Event-Dependent Actions - with and without Interlock

These actions are executed **once** when the event occurs and the corresponding step is active. When the sequencer is run through after this, an event-dependent action is executed again only if the event reoccurs.

S1: Actions linked to a step becoming active

Action				Explanation	Address range
Event	Instruction	Address	Location		
S1	N	Q, I, M, D*	m.n	As soon as the step becomes active (enters state), the signal state of the address is 1.	0.0 to 65535.7
S1	S	Q, I, M, D	m.n	As soon as the step becomes active (enters state), the address is set to 1 and then remains set to 1.	0.0 to 65535.7
S1	R	Q, I, M, D*	m.n	As soon as the step becomes active (enters state), the address is set to 0 and then remains set to 0.	0.0 to 65535.7
S1	CALL	FB, FC, SFB, SFC	Block number	As soon as the step becomes active (enters state), the specified block is called.	
S1	ON	S	I	As soon as the step becomes active (enters state), step i is activated.	i = step number
S1	OFF	S	I	As soon as the step becomes active (enters state), step i is deactivated.	i = step number
S1	OFF	S_ALL		As soon as the step becomes active (enters state), all steps are deactivated except for the step in which the action is located.	
S1	NC	Q,I,M,D	m.n	As soon as the step becomes active (enters state) and the condition (interlock) is satisfied, the signal state of the address is 1.	0.0 to 65535.7
S1	SC	Q, I, M, D*	m.n	As soon as the step becomes active (enters state) and the condition (interlock) is satisfied, the address is set to 1 and then remains set to 1.	0.0 to 65535.7

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
S1	RC	Q, I, M, D*	m.n	As soon as the step becomes active (enters state) and the condition (interlock) is satisfied, the address is set to 0 and then remains set to 0.	0.0 to 65535.7
S1	CALL C	FB, FC, SFB, SFC	Block number	As soon as the step becomes active and the condition (interlock) is satisfied, the specified block is called.	
S1	ONC	S	I	As soon as the step becomes active (enters state) and the condition (interlock) is satisfied, step i is activated.	i = step number
S1	OFFC	S	I	As soon as the step becomes active (enters state) and the condition (interlock) is satisfied, step i is deactivated.	i = step number
S1	OFFC	S_ALL		As soon as the step becomes active (enters state) and the condition (interlock) is satisfied, all steps are deactivated except for the step in which the action is located.	

S0: Actions linked to a step being deactivated

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
S0	N	Q, I, M, D*	m.n	As soon as the step is deactivated (leaves state), the signal state of the address is 1.	0.0 to 65535.7
S0	S	Q, I, M, D*	m.n	As soon as the step is deactivated (leaves state), the address is set to 1 and then remains set to 1.	0.0 to 65535.7
S0	R	Q, I, M, D*	m.n	As soon as the step is deactivated (leaves state), the address is set to 0 and then remains set to 0.	0.0 to 65535.7
S0	CALL	FB, FC, SFB, SFC	Block number	As soon as the step is deactivated (leaves state), the specified block is called.	
S0	ON	S	I	As soon as the step is deactivated (leaves state), step i is activated.	i = step number
S0	OFF	S	I	As soon as the step is deactivated (leaves state), step i is also deactivated.	i = step number

V1: Actions linked to supervision errors entering the state

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
V1	N	Q,I,M,D	m.n	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active, the signal state of the address is 1.	0.0 to 65535.7
V1	S	Q,I,M,D	m.n	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active, the address is set to 1 and then remains set to 1.	0.0 to 65535.7
V1	R	Q,I,M,D	m.n	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active, the address is set to 0 and then remains set to 0.	0.0 to 65535.7
V1	CALL	FB, FC, SFB, SFC	Block number	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active, the specified block is called.	
V1	ON	S	I	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active, step i is activated.	0.0 to 65535.7
V1	OFF	S	I	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active, step i is deactivated.	0.0 to 65535.7
V1	OFF	S_ALL		As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active, all steps are deactivated except for the step in which the action is located.	0.0 to 65535.7
V1	NC	Q,I,M,D	Mn	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active and the condition (interlock) is satisfied, the signal state of the address is 1.	0.0 to 65535.7
V1	SC	Q,I,M,D	m.n	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active and the condition (interlock) is satisfied, the address is set to 1 and then remains set to 1.	0.0 to 65535.7
V1	RC	Q,I,M,D	m.n	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active and the condition (interlock) is satisfied, the address is set to 0 and then remains set to 0.	0.0 to 65535.7

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
V1	CALLC	FB, FC, SFB, SFC	Block number	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active and the condition (interlock) is satisfied, the specified block is called.	
V1	ONC	S	I	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active and the condition (interlock) is satisfied, step i is activated.	0.0 to 65535.7
V1	OFFC	S	I	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active and the condition (interlock) is satisfied, step i is deactivated.	0.0 to 65535.7
V1	OFFC	S_ALL		As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active and the condition (interlock) is satisfied, all steps are deactivated except for the step in which the action is located.	0.0 to 65535.7

V0: Actions linked with supervision errors leaving the state

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
V0	N	Q,I,M,D	m.n	As soon as a supervision error leaves the state (is eliminated or acknowledged), the signal state of the address is 1.	0.0 to 65535.7
V0	S	Q,I,M,D	m.n	As soon as a supervision error leaves the state (is eliminated or acknowledged), the address is set to 1 and then remains set to 1.	0.0 to 65535.7
V0	R	Q,I,M,D	m.n	As soon as a supervision error leaves the state (is eliminated or acknowledged), the address is set to 0 and then remains set to 0.	0.0 to 65535.7
V0	CALL	FB, FC, SFB, SFC			
V0	ON	S	I	As soon as a supervision error leaves the state (is eliminated or acknowledged), step i is activated.	i = step number
V0	OFF	S	I	As soon as a supervision error leaves the state (is eliminated or acknowledged), step i is also deactivated.	i = step number

L0: Actions linked with interlocks entering the state

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
L0	N	Q,I,M,D	m.n	As soon as the interlock condition is satisfied (enters state) when the step is active, the signal state of the address is 1.	0.0 to 65535.7
L0	S	Q,I,M,D	m.n	As soon as the interlock condition is satisfied when the step is active (enters state), the address is set to 1 and then remains set to 1.	0.0 to 65535.7
L0	R	Q,I,M,D	m.n	As soon as the interlock condition is satisfied when the step is active (enters state), the address is set to 0 and then remains set to 0.	0.0 to 65535.7
L0	CALL	FB, FC, SFB, SFC	Block number	As soon as the interlock condition is satisfied when the step is active (enters state), the specified block is called.	
L0	ON	S	I	As soon as the interlock condition is satisfied when the step is active (enters state), step i is activated.	i = step number
L0	OFF	S	I	As soon as the interlock condition is satisfied when the step is active (enters state), step i is deactivated.	i = step number

L1: Actions linked with interlocks leaving state

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
L1	N	Q,I,M,D	m.n	As soon as the interlock condition is no longer satisfied (leaves state) while the step is active or if the interlock condition is not satisfied when the step becomes active, the signal state of the address is 1.	0.0 to 65535.7
L1	S	Q,I,M,D	m.n	As soon as the interlock condition is satisfied when the step is no longer satisfied (leaves state) while the step is active or if the interlock condition is not satisfied when the step becomes active, the address is set to 1 and then remains set to 1.	0.0 to 65535.7
L1	R	Q,I,M,D	m.n	As soon as the interlock condition is satisfied when the step is no longer satisfied (leaves state) while the step is active or if the interlock condition is not satisfied when the step becomes active, the address is set to 0 and then remains set to 0.	0.0 to 65535.7
L1	CALL	FB, FC, SFB, SFC	Block number	As soon as the interlock condition is satisfied when the step is no longer satisfied (leaves state) while the step is active or if the interlock condition is not satisfied when the step becomes active, the specified block is called.	
L1	ON	S	I	As soon as the interlock condition is satisfied when the step is no longer satisfied (leaves state) while the step is active or if the interlock condition is not satisfied when the step becomes active, step i is activated.	i = step number

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
L1	OFF	S	I	As soon as the interlock condition is satisfied when the step is no longer satisfied (leaves state) while the step is active or if the interlock condition is not satisfied when the step becomes active, step i is also deactivated.	i = step number
L1	OFF	S_ALL		As soon as the interlock condition is satisfied when the step is no longer satisfied (leaves state) while the step is active or if the interlock condition is not satisfied when the step becomes active, all steps are deactivated except for the step in which the action is located.	

A1: Actions linked to a message acknowledgment

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
A1	N	Q,I,M,D	m.n	If a message is acknowledged, the signal state of the address is 1 .	0.0 to 65535.7
A1	S	Q,I,M,D	m.n	As soon as a message is acknowledged, the address is set to 1 and then remains set to 1.	0.0 to 65535.7
A1	R	Q,I,M,D	m.n	As soon as a message is acknowledged, the address is set to 0 and then remains set to 0.	0.0 to 65535.7
A1	CALL	FB, FC, SFB, SFC	Block number	As soon as a message is acknowledged, the specified block is called.	
A1	ON	S	I	As soon as a message is acknowledged, step i is activated.	i = step number
A1	OFF	S	I	As soon as a message is acknowledged, step i is deactivated.	i = step number
A1	NC	Q,I,M,D	m.n	As soon as a message is acknowledged and the condition (interlock) is satisfied, the address has signal state 1.	0.0 to 65535.7
A1	SC	Q,I,M,D	m.n	As soon as a message is acknowledged and the condition (interlock) is satisfied, the address is set to 1 and then remains set to 1.	0.0 to 65535.7
A1	RC	Q,I,M,D	m.n	As soon as a message is acknowledged and the condition (interlock) is satisfied, the address is set to 0 and then remains set to 0.	0.0 to 65535.7
A1	CALLC	FB, FC, SFB, SFC	Block number	As soon as a message is acknowledged and the condition (interlock) is satisfied, the specified block is called.	
A1	ONC	S	I	As soon as a message is acknowledged and the condition (interlock) is satisfied, step i is activated.	i = step number
A1	OFFC	S	I	As soon as a message is acknowledged and the condition (interlock) is satisfied, step i is deactivated.	i = step number

R1: Actions linked to setting a registration

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
R1	N	Q,I,M,D	m.n	As soon as a registration is set, the address has signal state = 1.	0.0 to 65535.7
R1	S	Q,I,M,D	m.n	As soon as a registration is set, the address is set to 1 and then remains set to 1.	0.0 to 65535.7
R1	R	Q,I,M,D	m.n	As soon as a registration is set, the address is set to 0 and then remains set to 0.	0.0 to 65535.7
R1	CALL	FB, FC, SFB, SFC	Block number	As soon as a registration is set, the specified block is called.	
R1	ON	S	I	As soon as a registration is set, step i is activated.	i = step number
R1	OFF	S	I	As soon as a registration is set, step i is deactivated.	i = step number
R1	NC	Q,I,M,D	m.n	As soon as a registration is set and the condition (interlock) is satisfied, the signal state of the address is 1.	0.0 to 65535.7
R1	SC	Q,I,M,D	m.n	As soon as a registration is set and the condition (interlock) is satisfied, the address is set to 1 and then remains set to 1.	0.0 to 65535.7
R1	RC	Q,I,M,D	m.n	As soon as a registration is set and the condition (interlock) is satisfied, the address is set to 0 and then remains set to 0.	0.0 to 65535.7
R1	CALL C	FB, FC, SFB, SFC	Block number	As soon as a registration is set and the condition (interlock) is satisfied, the specified block is called.	
R1	ONC	S	I	As soon as a registration is set and the condition (interlock) is satisfied, step i is activated.	i = step number
R1	OFFC	S	I	As soon as a registration is set and the condition (interlock) is satisfied, step i is deactivated.	i = step number

Address D: Format: DBi.DBXm.n (i = block number, m = byte address, n = bit address)
For CALL[C] FB/SFB, an instance DB is required.

13.3 Counters in Actions

These actions are executed **once** when the event occurs and the corresponding step is active. When the sequencer is run through after this, an event-dependent action is executed again only if the event reoccurs.

S1: Counters in actions linked with activation of a step

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
S1	CS	C <initial counter value>	X	As soon as the step becomes active (enters state), the initial counter value is loaded in the counter.	x = counter number
S1	CU	C	X	As soon as the step becomes active (enters state), the counter is incremented by 1.	x = counter number
S1	CD	C	X	As soon as the step becomes active (enters state), the counter is decremented by 1.	x = counter number
S1	CR	C	X	As soon as the step becomes active (enters state), the counter is reset to 0.	x = counter number
S1	CSC	C <initial counter value>	X	As soon as the step becomes active (enters state) and the condition (interlock) is satisfied, the initial counter value is loaded in the counter.	x = counter number
S1	CUC	C	X	As soon as the step becomes active (enters state) and the condition (interlock) is satisfied, the counter is incremented by 1.	x = counter number
S1	CDC	C	X	As soon as the step becomes active (enters state) and the condition (interlock) is satisfied, the counter is decremented by 1.	x = counter number
S1	CRC	C	X	As soon as the step becomes active (enters state) and the condition (interlock) is satisfied, the counter is reset to 0.	x = counter number

S0: Counters in actions linked with deactivation of a step

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
S0	CS	C <initial counter value>	X	As soon as the step is deactivated (leaves state), the initial counter value is loaded in the counter.	x = counter number
S0	CU	C	X	As soon as the step is deactivated (leaves state), the counter is incremented by 1.	x = counter number
S0	CD	C	X	As soon as the step is deactivated (leaves state), the counter is decremented by 1.	x = counter number
S0	CR	C	X	As soon as the step is deactivated (leaves state), the counter is reset to 0.	x = counter number

L1: Counters in actions linked with interlocks leaving state

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
L1	CS	C <initial counter value>	X	As soon as the interlock condition is no longer satisfied (leaves state) while the step is active or if the interlock condition is not satisfied when the step becomes active, the initial counter value is loaded in the counter.	x = counter number
L1	CU	C	X	As soon as the interlock condition is no longer satisfied (leaves state) while the step is active or if the interlock condition is not satisfied when the step becomes active, the counter is incremented by 1.	x = counter number
L1	CD	C	X	As soon as the interlock condition is no longer satisfied (leaves state) while the step is active or if the interlock condition is not satisfied when the step becomes active, the counter is decremented by 1.	x = counter number
L1	CR	C	X	As soon as the interlock condition is no longer satisfied (leaves state) while the step is active or if the interlock condition is not satisfied when the step becomes active, the counter is reset to 0.	x = counter number

L0: Counters in actions linked with interlocks entering state

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
L0	CS	C <initial counter value>	X	As soon as the interlock condition is satisfied when the step is active (enters state), the initial counter value is loaded in the counter.	x = counter number
L0	CU	C	X	As soon as the interlock condition is satisfied when the step is active (enters state), the counter is incremented by 1.	x = counter number
L0	CD	C	X	As soon as the interlock condition is satisfied when the step is active (enters state), the counter is decremented by 1.	x = counter number
L0	CR	C	X	As soon as the interlock condition is satisfied when the step is active (enters state), the counter is reset to 0.	x = counter number

V1: Counters in actions linked to supervision errors entering the state

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
V1	CS	C <initial counter value>	X	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active, the initial counter value is loaded in the counter.	x = counter number
V1	CU	C	X	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active, the counter is incremented by 1.	x = counter number
V1	CD	C	X	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active, the counter is decremented by 1.	x = counter number
V1	CR	C	X	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active, the counter is reset to 0.	x = counter number
V1	CSC	C <initial counter value>	X	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active and the condition (interlock) is satisfied, the initial counter value is loaded in the counter.	x = counter number
V1	CUC	C	X	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active and the condition (interlock) is satisfied, the counter is incremented by 1.	x = counter number

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
V1	CDC	C	X	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active and the condition (interlock) is satisfied, the counter is decremented by 1.	x = counter number
V1	CRC	C	X	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active and the condition (interlock) is satisfied, the counter is reset to 0.	x = counter number

V0: Counters in actions linked to supervision errors leaving the state

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
V0	CS	C <initial counter value>	X	As soon as a supervision error leaves the state (is eliminated or acknowledged), the initial counter value is loaded in the counter.	x = counter number
V0	CU	C	X	As soon as a supervision error leaves the state (is eliminated or acknowledged), the counter is incremented by 1.	x = counter number
V0	CD	C	X	As soon as a supervision error leaves the state (is eliminated or acknowledged), the counter is decremented by 1.	x = counter number
V0	CR	C	X	As soon as a supervision error leaves the state (is eliminated or acknowledged), the counter is reset to 0.	x = counter number

A1: Counters in actions linked with a message acknowledgment

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
A1	CS	C <initial counter value>	X	As soon as a message is acknowledged), the initial counter value is loaded in the counter.	x = counter number
A1	CU	C	X	As soon as a message is acknowledged, the counter is incremented by 1.	x = counter number
A1	CD	C	X	As soon as a message is acknowledged, the counter is decremented by 1.	x = counter number
A1	CR	C	X	As soon as a message is acknowledged, the counter is reset to 0.	x = counter number
A1	CSC	C <initial counter value>	X	As soon as a message is acknowledged and the condition (interlock) is satisfied, the initial counter value is loaded in the counter.	x = counter number
A1	CUC	C	X	As soon a message is acknowledged and the condition (interlock) is satisfied, the counter is incremented by 1.	x = counter number
A1	CDC	C	X	As soon a message is acknowledged and the condition (interlock) is satisfied, the counter is decremented by 1.	x = counter number
A1	CRC	C	X	As soon as a message is acknowledged and the condition (interlock) is satisfied, the counter is reset to 0.	x = counter number

R1: Counters in actions linked to setting a registration

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
R1	CS	C <initial counter value>	X	As soon as a registration is set, the initial counter value is loaded in the counter.	x = counter number
R1	CU	C	X	As soon as a registration is set, the counter is incremented by 1.	x = counter number
R1	CD	C	X	As soon as a registration is set, the counter is decremented by 1.	x = counter number
R1	CR	C	X	As soon as a registration is set, the counter is reset to 0.	x = counter number
R1	CSC	C <initial counter value>	X	As soon as a registration is set and the condition (interlock) is satisfied, the initial counter value is loaded in the counter.	x = counter number

Action				Explanation	Address range
Event	Instruction	Address	Location		
R1	CUC	C	X	As soon a registration is set and the condition (interlock) is satisfied, the counter is incremented by 1.	x = counter number
R1	CDC	C	X	As soon a registration is set and the condition (interlock) is satisfied, the counter is decremented by 1.	x = counter number
R1	CRC	C	X	As soon as a registration is set and the condition (interlock) is satisfied, the counter is reset to 0.	x = counter number

13.4 Timers in Actions

These actions are executed **once** when the event occurs and the corresponding step is active. When the sequencer is run through after this, an event-dependent action is executed again only if the event reoccurs.

S1: Timers in actions linked with activation of a step

Action				Explanation	Address range
Event	Instruction	Address	Location		
S1	TL	T <time>	X	As soon as the step becomes active (enters state) the timer starts. For the specified time, the timer bit (status of the timer) is set to 1, when the time elapses, it is reset to 0.	x = no. of timer
S1	TD	T <time>	X	As soon as the step becomes active (enters state) the timer starts. For the specified time, the timer bit (status of the timer) is set to 0, when the time elapses, it is set to 1.	x = no. of timer
S1	TR	T	X	As soon as the step becomes active (enters state) the timer stops. Timer bit (status of the timer) and timer value are reset to 0.	x = no. of timer
S1	TLC	T <time>	X	As soon as the step becomes active and the condition (interlock) is satisfied, the timer starts. For the specified time, the timer bit (status of the timer) is set to 1, when the time elapses, it is reset to 0.	x = no. of timer
S1	TDC	T <time>	X	As soon as the step becomes active and the condition (interlock) is satisfied, the timer starts. For the specified time, the timer bit (status of the timer) is set to 0, when the time elapses, it is set to 1.	x = no. of timer
S1	TRC	T	X	As soon as the step becomes active and the condition (interlock) is satisfied, the timer stops. Timer bit (status of the timer) and timer value are reset to 0.	x = no. of timer

S0: Timers in actions linked with deactivation of a step

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
S0	TL	T <time>	X	As soon as the step is deactivated (leaves state) the timer starts. For the specified time, the timer bit (status of the timer) is set to 1, when the time elapses, it is reset to 0.	x = no. of timer
S0	TD	T <time>	X	As soon as the step is deactivated (leaves state) the timer starts. For the specified time, the timer bit (status of the timer) is set to 0, when the time elapses, it is set to 1.	x = no. of timer
S0	TR	T	X	As soon as the step is deactivated (leaves state) the timer stops. Timer bit (status of the timer) and timer value are reset to 0.	x = no. of timer

L1: Timers in Actions linked with interlocks leaving state

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
L1	TL	T <time>	X	As soon as the interlock condition is satisfied when the step is no longer satisfied (leaves state) while the step is active or if the interlock condition is not satisfied when the step becomes active, the timer starts. For the specified time, the timer bit (status of the timer) is set to 1, when the time elapses, it is reset to 0.	x = no. of timer
L1	TD	T <time>	X	As soon as the interlock condition is satisfied when the step is no longer satisfied (leaves state) while the step is active or if the interlock condition is not satisfied when the step becomes active, the timer starts. For the specified time, the timer bit (status of the timer) is set to 0, when the time elapses, it is set to 1.	x = no. of timer
L1	TR	T	X	As soon as the interlock condition is satisfied when the step is no longer satisfied (leaves state) while the step is active or if the interlock condition is not satisfied when the step becomes active, the timer stops. Timer bit (status of the timer) and timer value are reset to 0.	x = no. of timer

L0: Timers in actions linked with interlocks entering state

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
L0	TL	T <time>	X	As soon as the interlock condition is satisfied when the step is active (enters state), the timer starts. For the specified time, the timer bit (status of the timer) is set to 1, when the time elapses, it is reset to 0.	x = no. of timer
L0	TD	T <time>	X	As soon as the interlock condition is satisfied when the step is active (enters state), the timer starts. For the specified time, the timer bit (status of the timer) is set to 0, when the time elapses, it is set to 1.	x = no. of timer
L0	TR	T	X	As soon as the interlock condition is satisfied when the step is active (enters state), the timer stops. Timer bit (status of the timer) and timer value are reset to 0.	x = no. of timer

V1: Timers in actions linked to supervision errors entering the state

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
V1	TL	T <time>	X	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active, the timer starts. For the specified time, the timer bit (status of the timer) is set to 1, when the time elapses, it is reset to 0.	x = no. of timer
V1	TD	T <time>	X	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active, the timer starts. For the specified time, the timer bit (status of the timer) is set to 0, when the time elapses, it is set to 1.	x = no. of timer
V1	TR	T	X	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active, the timer stops. Timer bit (status of the timer) and timer value are reset to 0.	x = no. of timer
V1	TLC	T <time>	X	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active and the condition (interlock) is satisfied, the timer starts. For the specified time, the timer bit (status of the timer) is set to 1, when the time elapses, it is reset to 0.	x = no. of timer

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
V1	TDC	T <time>	X	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active and the condition (interlock) is satisfied, the timer starts. For the specified time, the timer bit (status of the timer) is set to 0, when the time elapses, it is set to 1.	x = no. of timer
V1	TRC	T	X	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active and the condition (interlock) is satisfied, the timer stops. Timer bit (status of the timer) and timer value are reset to 0.	x = no. of timer

V0: Timers in Actions linked to supervision errors leaving the state

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
V0	TL	T <time>	X	As soon as a supervision error leaves the state (is eliminated or acknowledged), the timer starts. For the specified time, the timer bit (status of the timer) is set to 1, when the time elapses, it is reset to 0.	x = no. of timer
V0	TD	T <time>	X	As soon as a supervision error leaves the state (is eliminated or acknowledged), the timer starts. For the specified time, the timer bit (status of the timer) is set to 0, when the time elapses, it is set to 1.	x = no. of timer
V0	TR	T	X	As soon as a supervision error leaves the state (is eliminated or acknowledged), the timer stops. Timer bit (status of the timer) and timer value are reset to 0.	x = no. of timer

A1: Timers in actions linked with a message acknowledgment

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
A1	TL	T <time>	X	As soon as a message is acknowledged, the timer starts. For the specified time, the timer bit (status of the timer) is set to 1, when the time elapses, it is reset to 0.	x = no. of timer
A1	TD	T <time>	X	As soon as a message is acknowledged, the timer starts. For the specified time, the timer bit (status of the timer) is set to 0, when the time elapses, it is set to 1.	x = no. of timer
A1	TR	T	X	As soon as a message is acknowledged, the timer stops. Timer bit (status of the timer) and timer value are reset to 0.	x = no. of timer
A1	TLC	T <time>	X	As soon as a message is acknowledged and the condition (interlock) is satisfied, the timer starts. For the specified time, the timer bit (status of the timer) is set to 1, when the time elapses, it is reset to 0.	x = no. of timer
A1	TDC	T <time>	X	As soon as a message is acknowledged and the condition (interlock) is satisfied, the timer starts. For the specified time, the timer bit (status of the timer) is set to 0, when the time elapses, it is set to 1.	x = no. of timer
A1	TRC	T	X	As soon as a message is acknowledged and the condition (interlock) is satisfied, the timer stops. Timer bit (status of the timer) and timer value are reset to 0.	x = no. of timer

R1: Timers in actions linked to setting a registration

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
R1	TL	T <time>	X	As soon as a registration is set, the timer starts. For the specified time, the timer bit (status of the timer) is set to 1, when the time elapses, it is reset to 0.	x = no. of timer
R1	TD	T <time>	X	As soon as a registration is set, the timer starts. For the specified time, the timer bit (status of the timer) is set to 0, when the time elapses, it is set to 1.	x = no. of timer
R1	TR	T	X	As soon as a registration is set, the timer stops. Timer bit (status of the timer) and timer value are reset to 0.	x = no. of timer

Action				Explanation	Address range
Event	**Instruction**	**Address**	**Location**		
R1	TLC	T <time>	X	As soon as a registration is set and the condition (interlock) is satisfied, the timer starts. For the specified time, the timer bit (status of the timer) is set to 1, when the time elapses, it is reset to 0.	x = no. of timer
R1	TDC	T <time>	X	As soon as a registration is set and the condition (interlock) is satisfied, the timer starts. For the specified time, the timer bit (status of the timer) is set to 0, when the time elapses, it is set to 1.	x = no. of timer
R1	TRC	T	X	As soon as a registration is set and the condition (interlock) is satisfied, the timer stops. Timer bit (status of the timer) and timer value are reset to 0.	x = no. of timer

13.5 Arithmetic in Actions

These actions are either executed in every cycle in which the relevant step is active or once when the relevant event occurs.

Action			Explanation
Event	**Instruction**	**Assignment**	
--	N	A:=B A:=func(B) A:=B<operator>C	As long as the step is active, the arithmetic is executed.
--	NC	A:=B A:=func(B) A:=B<operator>C	As long as the step is active and the condition (interlock) is satisfied, the arithmetic is executed.

S1: Arithmetic in actions linked with activation of a step

Action			Explanation
Event	**Instruction**	**Assignment**	
S1	N	A:=B A:=func(B) A:=B<operator>C	As soon as the step becomes active (enters state), the arithmetic is executed.
S1	NC	A:=B A:=func(B) A:=B<operator>C	As soon as the step becomes active (enters state) and the condition (interlock) is satisfied, the arithmetic is executed.

S0: Arithmetic in actions linked with deactivation of a step

Action			Explanation
Event	**Instruction**	**Assignment**	
S0	N	A:=B A:=func(B) A:=B<operator>C	As soon as the step is deactivated (leaves state), the arithmetic is executed.

L1: Arithmetic in Actions linked with interlocks leaving state

Action			Explanation
Event	**Instruction**	**Assignment**	
L1	N	A:=B A:=func(B) A:=B<operator>C	As soon as the interlock condition is no longer satisfied (leaves state) while the step is active or if the interlock condition is not satisfied when the step becomes active, the arithmetic is executed.

L0: Arithmetic in Actions linked with interlocks entering state

Action			Explanation
Event	**Instruction**	**Assignment**	
L0	N	A:=B A:=func(B) A:=B<operator>C	As soon as the interlock condition is satisfied (enters state) when the step is active, the arithmetic is executed.

V1: Arithmetic in actions linked to supervision errors entering the state

Action			Explanation
Event	**Instruction**	**Assignment**	
V1	N	A:=B A:=func(B) A:=B<operator>C	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active, the arithmetic is executed.
V1	NC	A:=B A:=func(B) A:=B<operator>C	As soon as a supervision error occurs while the step is active or a supervision error had been detected before the step became active and the condition (interlock) is satisfied, the arithmetic is executed.

V0: Arithmetic in actions linked to supervision errors leaving the state

Action			Explanation
Event	**Instruction**	**Assignment**	
V0	N	A:=B A:=func(B) A:=B<operator>C	As soon as a supervision error leaves the state (is eliminated or acknowledged), the arithmetic is executed.

A1:Arithmetic in actions linked with a message acknowledgment

Action			Explanation
Event	**Instruction**	**Assignment**	
A1	N	A:=B A:=func(B) A:=B<operator>C	As soon as a message is acknowledged, the arithmetic is executed.
A1	NC	A:=B A:=func(B) A:=B<operator>C	As soon as a message is acknowledged and the condition (interlock) is satisfied, the arithmetic is executed.

R1: Arithmetic in actions linked to setting a registration

Action			Explanation
Event	**Instruction**	**Assignment**	
R1	N	A:=B A:=func(B) A:=B<operator>C	As soon as a registration is set, the arithmetic is executed.
R1	NC	A:=B A:=func(B) A:=B<operator>C	As soon as a registration is set and the condition (interlock) is satisfied, the arithmetic is executed.

14 Configuration Instructions

Introduction

To automate your sequential control system with S7-GRAPH, in addition to the functions for the automatic mode, a range of further functions is also available. Before you can use the S7-GRAPH functions, you must make sure that certain conditions are met. To make it easier for you to get used to S7-GRAPH, the most important points are summarized in this chapter in the form of brief configuration instructions.

The following aspects are dealt with:

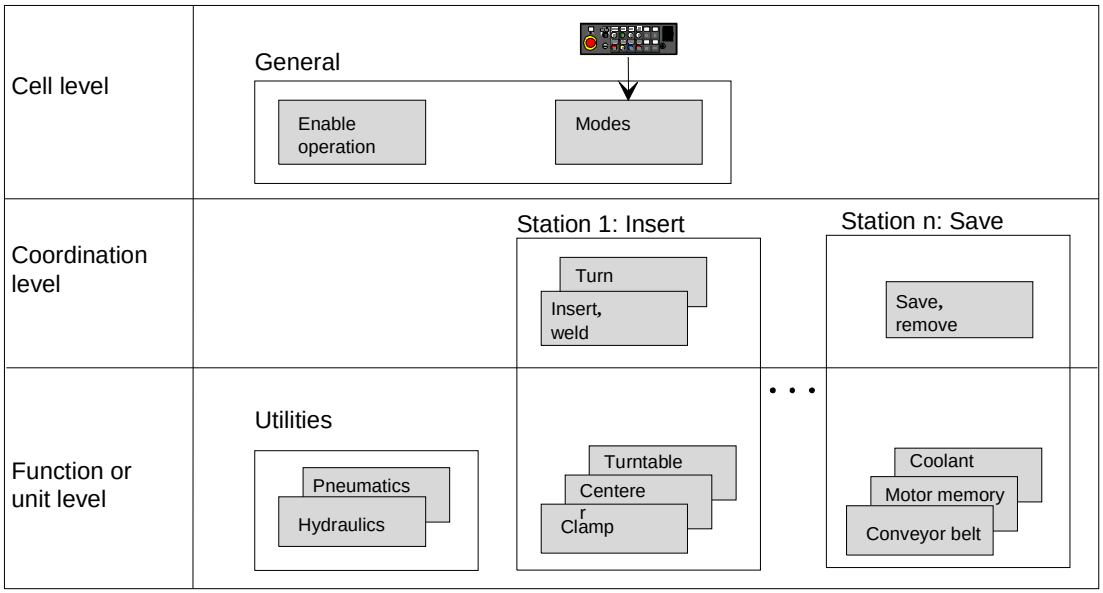
- Which general structures exist and how do these affect the program structure of the sequencer?
- What is important when programming and using sequencers?
- How are operator interface systems connected?

14.1 General Level and Program Structure

Level Structure Based on the Example of a Manufacturing Cell

When using S7-GRAPH, remember that the sequencer is designed primarily to configure sequential processes clearly and simply. Apart from these sequences with which, for example, the parts of a manufacturing unit can be coordinated, additional higher-level functions (cell level) and lower-level functions (function or unit level) are also required.

The sequencer programmed with S7-GRAPH belongs to the coordination level, but must take into account the interdependencies from the cell level and the unit level.

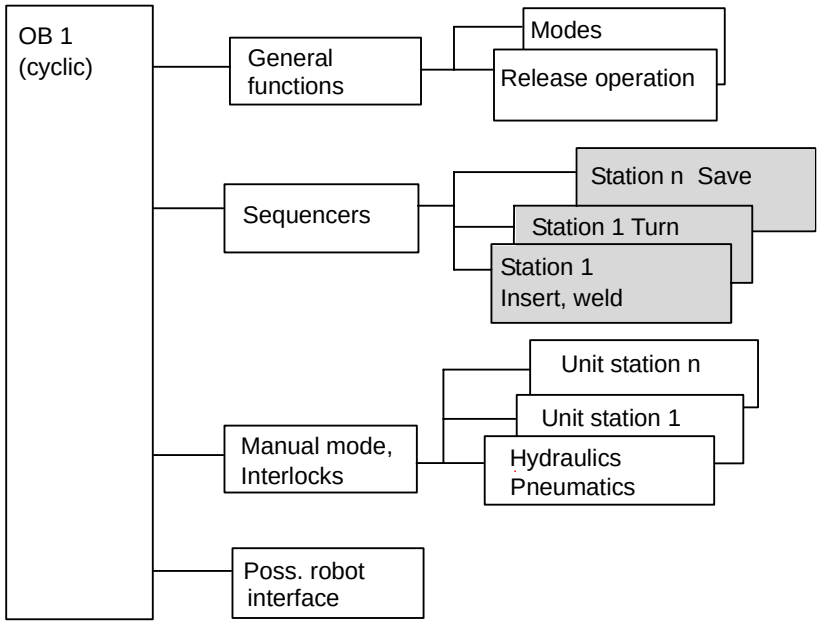


The individual levels have the following significance:

- Cell level
  The cell level encompasses functions with a central or wider reaching significance, for example preparations for enabling operation and modes. The blocks at the cell level provide signals that are relevant for all blocks of the coordination and unit level in this cell.
- Coordination level
  The coordination level includes various coordination functions, for example for the automatic mode and retraction following a break in operation. For each station, at least one sequencer is required for coordination at this level.
- Unit level
  The unit level includes functions for activating individual units of equipment, for example motors and valves. This includes all the lower-level functions, for example interlocks and supervision conditions independent of the sequencer and direct manual intervention in the operation of the units.

Specifying the Program Structure and Incorporating the Sequencer

For each sequencer, S7-GRAPH creates an FB with its instance DB. This S7-GRAPH FB must be called in a block (for example OB, FB or FC). Since other programs are normally required along with the programs created by S7-GRAPH, it is usually the best policy to call all the FBs created by F7 Graph in one block (FC or FB) as shown in the following example.



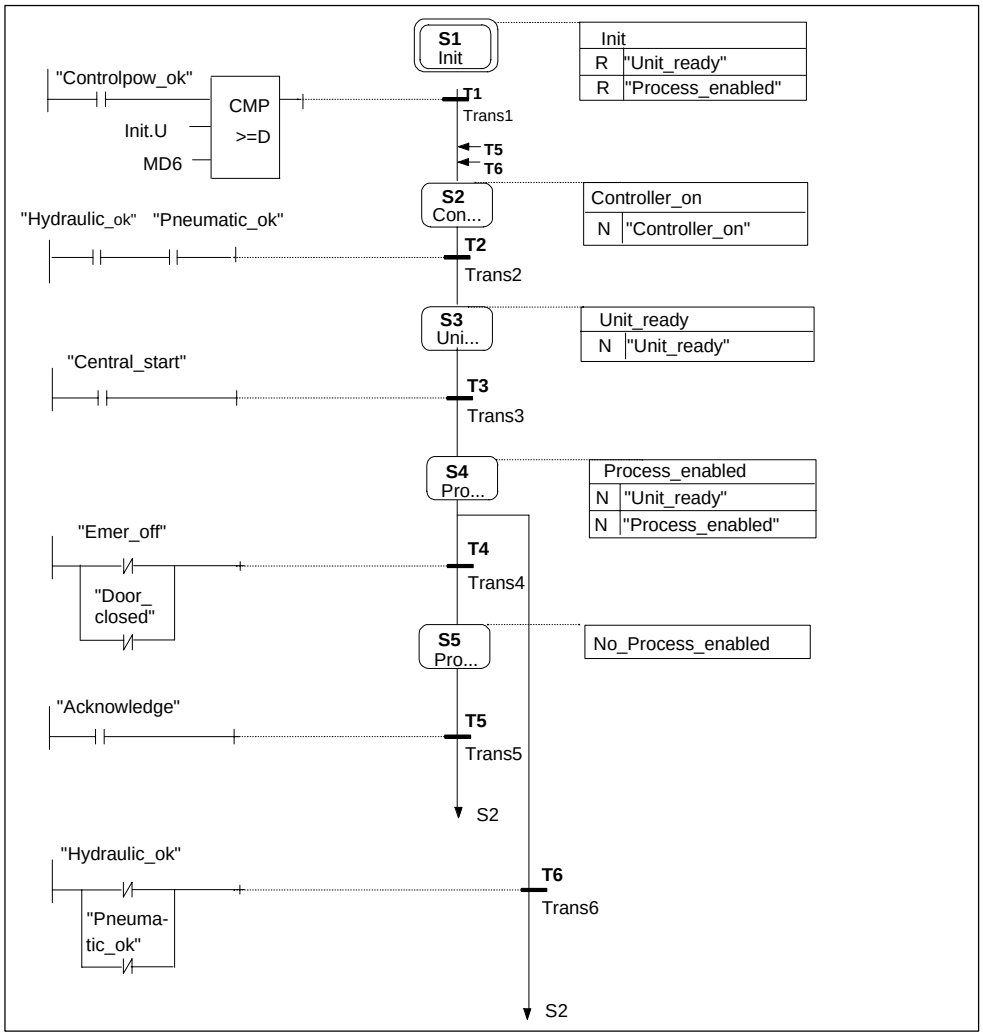
The various functions at the individual levels are executed cyclically in the order in which they are called:

- The sequencers are preceded by the central, higher-level functions of the cell level.
- The various sequencers for the individual stations are called by an FB "Sequencers" that is called in OB1.
- The program sections for manual mode, the interlocks and permanent monitoring functions for the units follow the sequencers.
- If the manufacturing cell includes robots, program sections for supplying and clearing robot interface are also required.

Enabling Operation

In production facilities, utilities such as hydraulic and pneumatic systems are often required and the functions of all other units depend on them being available. It therefore makes sense to turn on the hydraulic and pneumatic systems centrally after turning on the machinery and to generate ready signals for the other blocks.

The following diagram is an example of how general operation of the system can be enabled in S7-GRAPH.



- After turning on the controller, step **1** (initial step) is activated using the INIT_SQ parameter and the signals Unit_ready and Process_enabled are reset.
- After a specified minimum waiting time, **step 2** turns on the hydraulic and pneumatic units dependent on the signal Controlpow_ok (control voltage OK) using the Controller_on action.
- After the feedback indicating that the hydraulic and pneumatic systems are OK, **step 3** outputs the message Unit_ready (to various recipients including the central controller).

- Triggered by the signal "Central_start" (start main console of the production equipment), **step 4** sets the unit to the ready state and enables automatic execution.
- With "Emer_off" or loss of the signal "Door_closed", the change to **step 5** cancels the Process_enabled. If this is acknowledged, there is a jump to step 2.
- If a step enabling condition from one of the power systems is canceled while the Process_enabled (step 4) is active, the enable signals are also canceled with the change to step 2. Once the step enabling conditions ("OK_Signal") of the power systems are present again, Central_start can return the unit to the enabled state again.
- The signals "Controller_on", "Unit_ready" and "Process_enabled" are sent to the other blocks and the central controller.
- The "Unit_ready" signal also enables the mode selection in the mode block.
- The "Process_enabled" signal enables the automatic mode.

Handling Modes

Depending on the situation, the user requires different modes for the system and machinery. In unlinked manufacturing cells, for example in body work assembly, these are the modes:

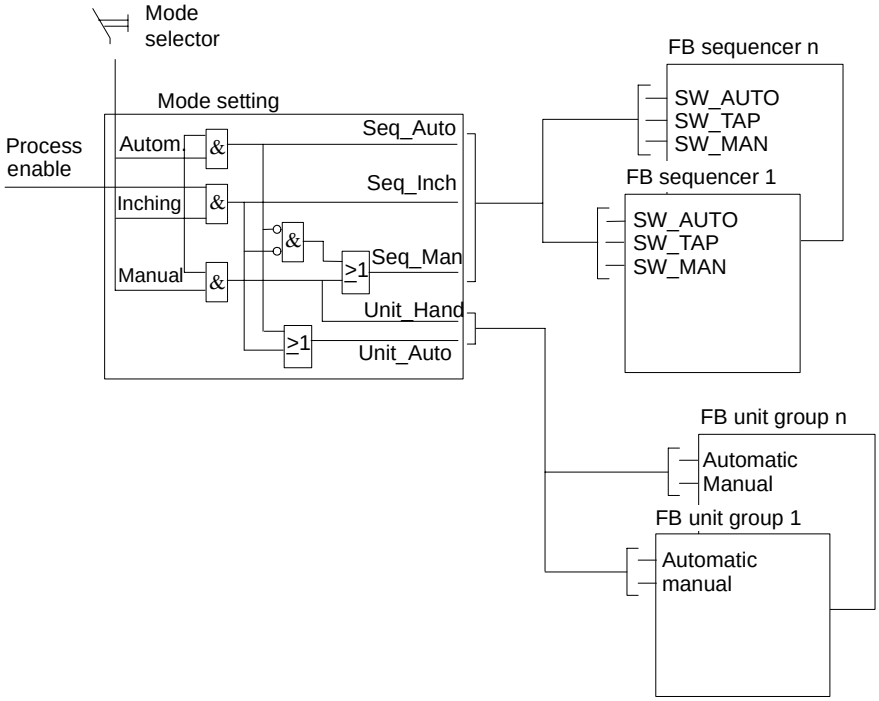
- Automatic
- Inching, in other words a variant of the automatic mode with a stop after each process step
- Automatic or switch to next; in other words the automatic mode with an additional step enabling condition
- Manual/setup, in other words direct influence on the units or functions

The modes, however, do not have the same effects at all levels. The following overview lists the modes and their effects at the coordination level and unit level (motors, valves etc.).

Mode	**Effect at the coordination level**	**Effect at the unit level**
Automatic	Sequencer passes control to next step when the transition is satisfied.	Enabling of the drive by the sequencer
Inching	The sequencer passes control to the next step when the transition is satisfied and the "Inching" button signal enables the next step.	Enabling of the drive by the sequencer
Automatic or step by step mode	The sequencer progresses when the transition is satisfied or the "Inching" button signal enables the next step.	Enabling of the drive by the sequencer
Manual or setup	Enabling of next step suppressed, the status of the sequencer has no effect at the unit level.	Drive enabled by direction buttons

Representing the Mode in the Sequencer

If, for example, a selector switch on the control panel supplies the signals automatic, inching, and manual as individual single signals, only relatively simple logic is required to derive the mode signals for the sequencers and for the underlying drives of the units as shown in principle in the following diagram.

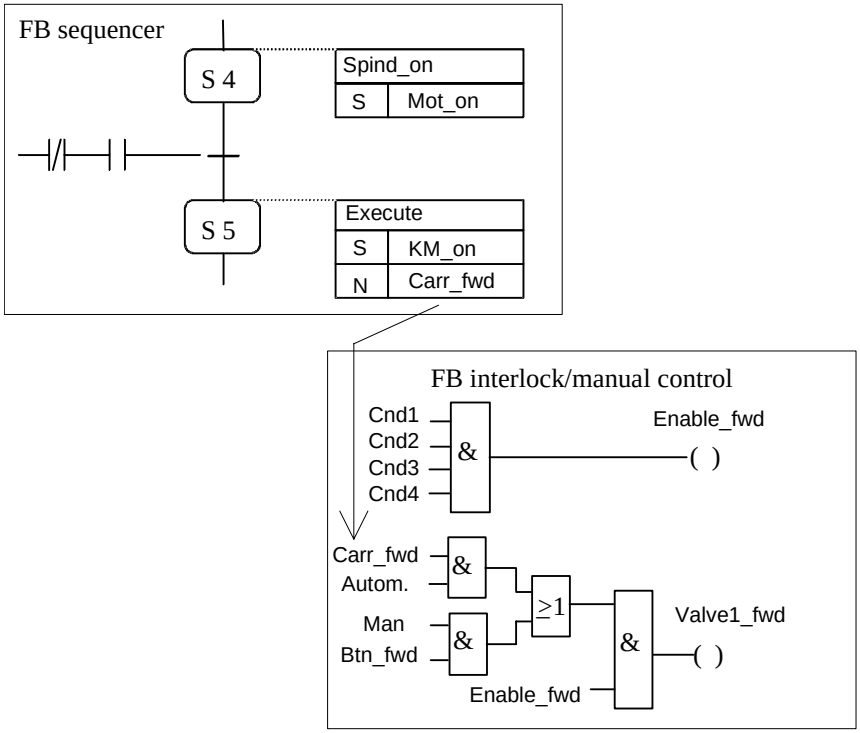


The modes selected at the console are passed on to the sequencers when Process_enabled is active. If Process_enabled is not set, the modes of the sequencers are switched to SW_MAN and step enabling is suppressed.

In the automatic or inching modes, the automatic mode is set for the unit functions. The manual mode is passed on directly. If there is no process_enabled, both modes are reset and the process is stopped. The functions of the unit groups are represented in an FB created with LAD/FBD or STL.

Handling the Interlocks and Manual Control in the Manual Mode

In S7-GRAPH, the sequencer controls the functions for the automatic mode. The functions for the manual mode required over and above those of the automatic mode are included in a separate FB. Using action bits (for example Carr_fwd) the sequencer is connected to the interlock and manual control.



The permanent monitoring functions required for the units can be programmed with S7 PDIAG. Depending on the requirements, the supervision conditions can be formulated so that they are effective both in the automatic and manual modes or specific supervision conditions are programmed for the automatic mode and for the manual mode.

14.2 Points to Note about Programming Sequencers

Initialization

The sequencer is initialized using the INIT_SQ parameter. If this changes to "1", all steps marked as initial steps are activated. All other steps are deactivated.

After turning on the controller, the sequencer is in the same mode as it was when the controller was turned off, in other words if the controller was in the SW_AUTO mode before it was turned off, it will be in this mode when it is turned on again. If you want to avoid this, the sequencer must be set to the required mode (for example SW_MAN) explicitly when it is turned on, for example using the initialization signal.

If the status of the process (system or machine) does not correspond to the initial state of the sequencer, you can deal with this in two ways:

- The process is, for example, changed to the initial state by a command.
  If you want to change the process to the initial state, it is usually advisable to create a separate sequencer for this purpose.
- The sequencer must be synchronized with the process state using the synchronize function, in other words, the sequencer must be set to the current part of the process.

Which method you use depends on the type of process.

Interlocks

In S7-GRAPH, interlocks can be defined for each step. All actions that are executed depending on a condition are only activated when the interlock conditions are satisfied.

If the interlock conditions are not satisfied when the step is active, an error message is sent to the console (for example operator panel). The error is cleared when the interlock conditions are satisfied.

Supervisions

In S7-GRAPH, monitoring conditions can be defined in the form of supervisions for each step. The most common supervision condition is the execution time of the actions. If the supervision condition is satisfied, in other words an error has occurred, an error message is sent to the console (for example operator panel). Such errors are acknowledged via the ACK_EF parameter.

The compilation settings (menu command **Options > Block Settings**) decide whether an acknowledgment is be required when an S7-GRAPH FB is created.

- Compilation options "Acknowledge Errors" checked:
  The error is cleared only when an acknowledgment signal is sent and the supervision is no longer satisfied or when an acknowledgment signal is sent and the successor transition is satisfied.

  If time monitoring was programmed with Si.U, the acknowledgment signal resets the monitoring time so that the supervision condition is no longer satisfied.
- Compilation options "Acknowledge Errors" not checked:
  The error is cleared when the supervision conditions are no longer satisfied.

Caution

If the step enabling condition is satisfied during the acknowledgment cycle, the next step is enabled.

Please remember that the supervisions defined in the sequencer are specific to a particular step and only monitor the execute time of the relevant actions. If, for example, a motor is turned on in step 1, the run-up time can be monitored in step 1. Once this step is deactivated, the motor is no longer monitored in the sequencer. If you want the motor to be monitored permanently and independent of the sequencer, this must implemented at the underlying unit level.

Abort Situations

Abort situations occur when an "immediate stop" or "emergency off" is set due to a critical situation or when a fault occurs in units. In these situations, the following operations are carried out:

- The sequencers are stopped by switching to SW_MAN.
- At the unit level, the output signals are deactivated by resetting the automatic enable.

Synchronization

Synchronization of the sequencer with the state of the process or machine is required at least in the following situations:

- Case 1: A program change in the installation and startup phase meant that the instance DB had to be regenerated.
- Case 2: In the manual mode, units were controlled manually so that the state of the sequencer no longer corresponds to the state of the process or machine. Following a manual intervention, and depending on the process, the automatic mode must be resumed starting at the current state of the system or machine; in other words, the current position of the sequencer must be adapted to the process (synchronized).

In Case 1, the menu command **Debug > Control Sequencer** can be used in the manual mode to set and reset steps directly when working on the programming device so that the state that existed before the program change can be set.

In Case 2, synchronization can be achieved directly working on the programming device. Synchronization then involves the following steps:

1. Select the menu command **Options > Block Settings** and make sure that the "Synchronization" option is selected in the "Compile/Save" tab.
2. Select the menu command **Debug > Synchronization**.

Result: All transition and interlock conditions are detected. The subsequent evaluation shows which steps are located between a satisfied and non-satisfied transition. All steps found are displayed yellow.

3. Click all the steps you want to select with the mouse pointer that changes to a cross-hair.
4. Select the steps with the "Activate" button.

In principle, you can also synchronize using a console (for example operator panel). This is possible when permanent execution of transitions and interlock conditions is triggered at the console and the results are displayed and required steps can be activated on the console.

Program-controlled synchronization is also possible, when, for example a button signal brings about a change from the initial step to synchronization in a "Synchronization" step in which process state transitions leaving the state are evaluated. Using jump commands, the sequencer can then jump to the suitable step.

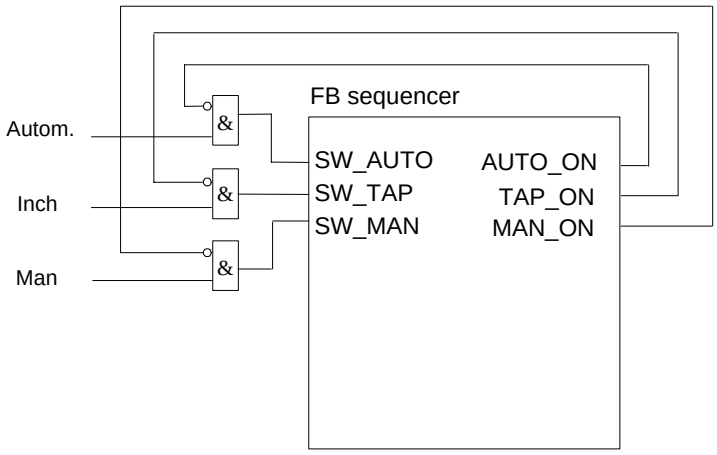
Modifications during Installation and Commissioning

If changes are made to the sequencer structure during installation and commissioning, for example a step is added, deleted, or renumbered or a transition is deleted or added, a new instance DB is created when you compile the sequencer. Downloading the instance DB to the PLC has the same effect as initialization with the INIT_SQ input, in other words all initial steps are active and all other steps are inactive. The actions are, however, not automatically invalidated. Download the instance DB only in the manual mode to avoid errors that can occur when the data are overwritten.

Using the menu command **Debug > Control Sequencer**, you can reset the sequencer to the process state that applied before the modification was made.

Caution

With the menu command **Debug > Control Sequencer**, you can change the mode of the sequencer directly in S7-GRAPH. If the sequencer and underlying monitoring/manual functions are supplied with parameters as described in Handling Modes, this means that the mode of the sequencer no longer matches the underlying function. If you want to prevent this, the mode change triggered on the programming device can be disabled. The following diagram illustrates the principle.

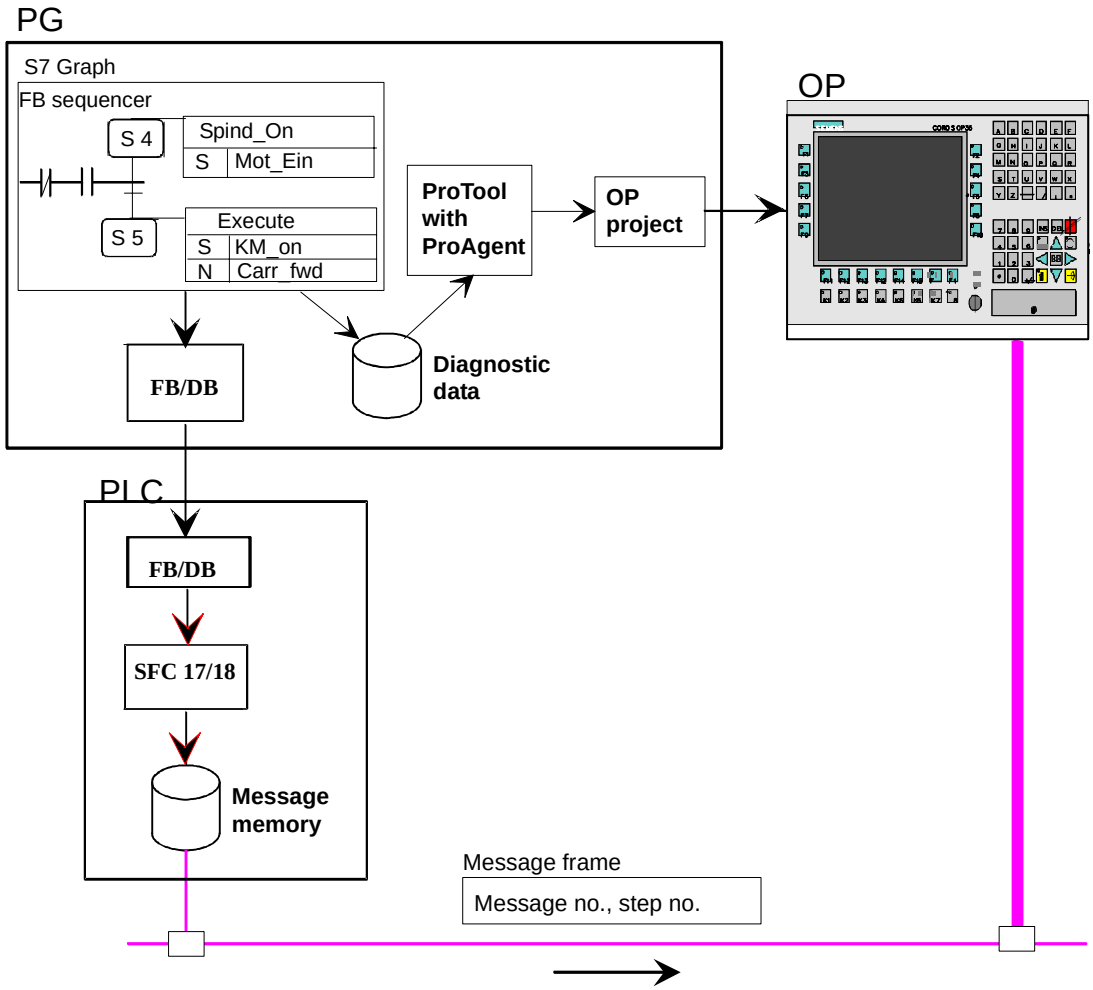


With this logic, the sequencer is reset to the state set by the signals automatic/inching/manual after one cycle.

14.3 Linking to Operator Control and Monitoring Systems

The task of a machine or system is to produce. During normal operation, the display of the process state is normally unimportant. The situation is different when the process is disturbed. In this case it is important to be able to recognize an eliminate the cause as quickly as possible. With the monitoring and interlock functions in S7-GRAPH and with the standardized diagnostic interfaces, this requirement is met ideally.

The following diagram shows the interaction between a programming device with S7-GRAPH, the programmable logic controller, and the operator panel.



Creating Diagnostic Data

When you compile a sequencer, you can decide whether diagnostic data should also be created in addition to the FB/DB providing the "message with ALARM_SQ/ALARM_S" option is selected. The diagnostic data contains the structure data of a sequencer and can be linked into a corresponding OP project using ProAgent.

This makes the following functions possible on the operator panel:

- Display of the current state of the sequencer with current sequencer names and step names
- Display of a disturbed step (step number and name)
- Display of the cause of a disturbance, in other words:
  - If a supervision error occurs: Display of the missing step enabling condition
  - If an interlock error occurs: Display of the missing interlock condition.

If you have modified addresses in the SIMATIC manager using the **Rewire** menu command, and the addresses are also used by the sequencer, the corresponding diagnostic data are only updated when you open the corresponding S7-GRAPH FB again and save it.

Creating Messages

To display disturbances on, for example, an operator panel, you must click the option "Message with ALARM_SQ/ALARM_S (SFC17/SFC18) in the "Messages" tab that you can display with the menu command **Options > Block Settings**. If this option is selected, a message is entered in the message memory of the CPU by the selected SFC and automatically transferred to the operator panels. The message frame contains not only the automatically assigned message number but also the current step number so that the operator panel can display the corresponding texts and depending on the display, the corresponding conditions.

Apart from the message, a group error bit is also set that is displayed on the operator panel depending on the display.

The following displays are available on the operator panel for the sequencer:

- Overview level
  All units are displayed with their current step. Each sequencer block represents such a unit. If a sequencer has a disturbance, this line is marked by  . I f this line is selected, you can change directly to the detailed display level with a function key.
- Message display level
  The current message with the date and time is displayed. You can change directly to the detailed display from this level.
- Detailed display
  The network affected is displayed in Ladder Logic or statement list representation and the missing conditions are marked. In the display, you also have the option of displaying the entire network or the reduced network containing only the conditions affected. This is particularly useful with more extensive logic operations.

Note

- If you use the standard FC71, you do not require the SFC17/SFC18 blocks. Messages are, however, processed using SFC52 ("WR_USMSG").
- If you use the standard FC73, no messages are generated due to the minimized memory configuration.

---

15 Instance DB

Each S7-GRAPH FB is assigned an instance data block that can be understood as representing the memory of the FB. It contains the data belonging to the sequencer, in other words all the data and parameters required to execute the FB:

- All the FB parameters of the selected FB parameter set
- step-transition description (for example current or last activation time of a step with and without disturbance times; indicator whether or not a step is active).

The data are entered automatically.

15.1 Settings for the Instance DB

Include Instance DB

To create the instance DB automatically after compiling an error-free FB, click the option "Include Instance DB" in the "General" tab (menu command **Options > Application Settings**). If you select this option, whenever you compile or download the block, S7-GRAPH checks whether it is necessary to create or update the instance DB and takes whatever action is necessary.

Interface Description

In the "Save/Compile" tab, there are three possible settings for the interface description of the instance DB:

- Memory minimized (as of V5.1)
  The interface descriptions are stored in the instance DB as structures. A separate structure with the essential information is created for each step and each transition of the sequencer. This option reduces the memory requirements of your S7-GRAPH FB considerably. The blocks created do not, however, have diagnostic capability. When you monitor the sequential control system, you see the status display for conditions for the active step only.
  If you use this option, you must use the supplied FC75 standard block.

Using Standard Function Blocks FC70, FC71, FC72 and FC73

- Structure Arrays
  The interface descriptions are stored in the instance DB as fields (ARRAYs). This option optimizes the memory requirements of your S7-GRAPH function block. The step names are, however, not stored. Symbolic addressing of other blocks is not possible in this case.
- Individual Structures

  The interface descriptions are stored in the instance DB as structures (STRUCT). A separate structure with detailed information is created for each step and each transition of the sequencer. This is a convenient method allowing internal and external access using symbolic names. The instance DB can be evaluated both using direct access to the data or using the step and transition names. The structures increase the size of the instance DB but do not affect the performance when executing the sequencer.

15.2 Assignment of Instance DB and S7-GRAPH FB

In general, S7-GRAPH makes an automatic DB assignment for blocks. If the DB with the same number as the FB does not yet exist or the DB and FB match, the DB is automatically assigned. In all other cases (even when compiling source files), you must enter the number of the DB. Follow the steps outlined below:

- Select the menu command **File > Modify/Adapt Instance DB Number** and enter a new DB name in the dialog box that is opened. An instance DB matching the FB is created.

Proposed DB Number when First Compiling a Source File

When an instance DB is created the first time (either automatically or manually), S7-GRAPH proposes a DB number in a dialog box that is the same as the number of the current FB. If your function block, for example, is called "FB6", S7-GRAPH proposes "DB6". You can either accept the proposed DB name or click the "Select" button and specify a different DB name or a different path.

15.3 Automatic Creation of the Instance DB on Saving

If you have set the option "Include Instance DB" in the "General" tab that you display with the menu command **Options > Block Settings** , follow the steps below:

- Program the S7-GRAPH FB and select the menu command **File > Save**.

  Result: The S7-GRAPH FB is saved and the instance DB is created.

Caution

The data block must not be created manually or deleted if you have set the option "message with ALARM_SQ / ALARM_S (SFC17 / SFC18)" in the "Messages" tab (menu command **Options > Block Settings**) otherwise messages can be lost. The instance data block is then included automatically.

15.4 Creating the Instance DB Later

If you have not selected the option "Include Instance DB" in the "General" tab, follow the steps below:

1. Program the S7-GRAPH FB and select the menu command **File > Save**.
2. If the S7-GRAPH FB was compiled free of errors, select the menu command **File > Create Instance DB**.

   Result: A new instance DB is created or the selected instance DB is overwritten.

15.5 Displaying and Printing the Content of the Instance DB

To display and print the content of the instance DB, follow the steps outlined below:

1. While in the SIMATIC Manager, open the DB in the relevant user program by double-clicking it.
2. Select the menu command **View > Data View** in the LAD/STL/FBD editor.
3. To print the content of the instance DB, select the menu command **File > Print**.

15.6 Accessing the Instance DB

You can access the instance DB internally, in other words within the S7-GRAPH FB (for example as an action linked to an event) or externally, in other words outside the S7-GRAPH FB (for example linked to a different block).

Access to the Data of the Instance DB

You can access the instance DB with STEP 7 tools. We strongly advise, however, that you do not edit the DB.

15.7 Structure of an Instance DB

The structure of the instance DB depends on the settings you make in the "Compile / Save" tab (menu command **Options > Block Settings**). The following factors affect the content and the size.

1. The parameter set selected for the FB
2. The interface description for the data belonging to steps and transitions (memory minimized/structure arrays/individual structures). The data are entered automatically.
3. The option of entering criteria analysis data.
4. Number and size of user-defined variables

15.8 Outline of the Structure of the Instance DB

The following table illustrates the divisions within the instance DB.

Area	Identifier	Length
FB parameters	Names of the input and output parameters of the S7-GRAPH FB	Depends on the selected parameter set: 2 bytes: minimum 10 bytes: Standard/Maximum Individual: Maximum V5/User-def.
Reserved work area	G7T_0	16 bytes
Transitions	Transition name (for example Trans1, Trans2)	(number of transitions) ∗ 16 bytes
Reserved work area	G7S_0	32 bytes
Steps	step name (for example Step1, Step2)	(number of steps) ∗ 32 bytes
Sequencer status	-	-
Internal work area	-	-

15.9 FB Parameters

The FB parameters are located in the instance DB. The space required by the FB parameters depends on the setting (minimum, standard, maximum or user-defined parameter set) made in the "Compile / Save" tab selected with the menu command **Options > Block Settings** . (If, for example, the minimum parameter set is selected, INIT_SQ is the only parameter displayed).

The following table shows the maximum parameter set V5 (definable).

Parameter	Description	Data Type	Read int.	Write int.	Read Ext.	Write Ext.
OFF_SQ	OFF_SEQUENCE: Turn off the sequencer	BOOL	yes	no	yes	no
INIT_SQ	INIT_SEQUENCE: Initialize sequencer	BOOL	yes	no	yes	no
ACK_EF	ACKNOWLEDGE_ERROR_FAULT: Acknowledge all errors, force switching to next step	BOOL	yes	no	yes	no
REG_EF	REGISTRATE_ERROR_FAULT: Register all errors and disturbances	BOOL	yes	no	yes	no
ACK_S	ACKNOWLEDGE_STEP Acknowledge step displayed in S_NO	BOOL	yes	no	yes	no
REG_S	REGISTRATE_STEP Register step displayed in S_NO	BOOL	yes	no	yes	no
HALT_SQ	HALT_SEQUENCE Stop/reactivate sequencer	BOOL	yes	no	yes	no
HALT_TM	HALT_TIMES: Stop/reactivate all step activation times and time-dependent instructions (L and D) of the sequencer	BOOL	yes	no	yes	no
ZERO_OP	ZERO_OPERANDS: Reset all addresses of the instructions N, D, L in active steps to zero and do not execute CALL instructions in actions/reactivate addresses and CALL instructions.	BOOL	yes	no	yes	no
EN_IL	ENABLE_INTERLOCKS: Deactivate/reactivate interlocks	BOOL	yes	no	yes	no
EN_SV	ENABLE_SUPERVISIONS: Deactivate/reactivate supervisions	BOOL	yes	no	yes	no
EN_ACKREQ	ENABLE_ACKNOWLEDGE_ REQUIRED: Activate mandatory acknowledgment	BOOL	yes	no	yes	no
EN_SSKIP	ENABLE_STEP _SKIPPING: Activate skip step	BOOL	yes	no	yes	no
DISP_SACT	DISPLAY_ACTIVE_STEPS: Display active steps only	BOOL	yes	no	yes	no
DISP_SEF	DISPLAY_STEPS_WITH_ERROR_ OR_FAULT: Display steps with errors and disturbed steps only	BOOL	yes	no	yes	no

Parameter	Description	Data Type	Read int.	Write int.	Read Ext.	Write Ext.
DISP_SALL	DISPLAY_ALL_STEPS: Display all steps	BOOL	yes	no	yes	no
S_PREV	PREVIOUS_STEP: Automatic mode: Indicate previous simultaneously active steps in S_NO Manual mode: Indicate previous step (next lower number)	BOOL	yes	no	yes	no
S_NEXT	NEXT_STEP: Automatic mode: Indicate next simultaneously active step in S_NO manual mode: Indicate next step (next higher number)	BOOL	yes	no	yes	no
SW_AUTO	SWITCH_MODE_AUTOMATIC: Mode changer: Automatic mode	BOOL	yes	no	yes	no
SW_TAP	SWITCH_MODE_TRANSITION_AND_PUSH: Mode changer: Inching mode ("semi-automatic")	BOOL	yes	no	yes	no
SW_TOP	SWITCH_MODE_TRANSITION_OR_PUSH: Mode change: Automatic or switch to next	BOOL	yes	no	yes	no
SW_MAN	SWITCH_MODE_MANUAL: Mode changer: Manual mode, no triggering of automatic execution	BOOL	yes	no	yes	no
S_SEL	STEP_SELECT: Selects a specific step for the output parameter S_NO. Activate/deactivate in the manual mode with S_ON, S_OFF.	INT	yes	no	yes	no
S_SELOK	STEP_SELECT_OK: Use value in S_SEL for S_NO	BOOL	yes	no	yes	no
S_ON	STEP_ON: Manual mode: Activate indicated step	BOOL	yes	no	yes	no
S_OFF	STEP_OFF: Manual mode: Deactivate the displayed step	BOOL	yes	no	yes	no
T_PREV	PREVIOUS_TRANSITION: Display previous valid transition in T_NO	BOOL	yes	no	yes	no
T_NEXT	NEXT_TRANSITION: Display next valid transition in T_NO	BOOL	yes	no	yes	no
T_PUSH	PUSH_TRANSITION: Transition passes control when the condition is satisfied and there is a rising edge at T_PUSH; condition: Inching mode (SW_TAP)	BOOL	yes	no	yes	no
S_NO	STEP_NUMBER Display step number	INT	yes	no	yes	no

Parameter	Description	Data Type	Read int.	Write int.	Read Ext.	Write Ext.
S_MORE	MORE_STEPS: Further active steps can be selected in S_NO	BOOL	yes	no	yes	no
S_ACTIVE	STEP_ACTIVE Displayed step is active	BOOL	yes	no	yes	no
S_TIME	STEP_TIME Step activation time	TIME	yes	no	yes	no
S_TIMEOK	STEP_TIME_OK: No error in step activation time	TIME	yes	no	yes	no
S_CRITLOC	STEP_CRITERIA_INTERLOCK Interlock criteria bits	DWORD	yes	no	yes	no
S_CRITLOC ERR	S_CRITERIA_IL_LAST_ERROR: Interlock criteria bits for event L1	DWORD	yes	no	yes	no
S_CRITSUP	STEP_CRITERIA_SUPERVISION: Supervision criteria bits	DWORD	yes	no	yes	no
S_STATE	STEP_STATE: Step state bits	WORD	yes	no	yes	no
T_NO	TRANSITION_NUMBER: Valid transition number	INT	yes	no	yes	no
T_MORE	MORE_TRANSITIONS: Other valid transitions available for display	BOOL	yes	no	yes	no
T_CRIT	TRANSITION_CRITERIA: Transition criteria bits	DWORD	yes	no	yes	no
T_CRITOLD	T_CRITERIA_LAST_CYCLE: Transition criteria bits from last cycle	DWORD	yes	no	yes	no
T_CRITFLT	T_CRITERIA_LAST_FAULT: Transition criteria bits for event V1	DWORD	yes	no	yes	no
ERROR	INTERLOCK_ERROR: Interlock error (any step)	BOOL	yes	no	yes	no
FAULT	SUPERVISION_FAULT: Supervision error (any step)	BOOL	yes	no	yes	no
ERR_FLT	IL_ERROR_OR_SV_FAULT: Group disturbance	BOOL	yes	no	yes	no
SQ_ISOFF	SEQUENCE_IS_OFF: Sequencer off	BOOL	yes	no	yes	no
SQ_HALTED	SEQUENCE_IS_HALTED: Sequencer stopped	BOOL	yes	no	yes	no
TM_HALTED	TIMES_ARE_HALTED: Timers stopped	BOOL	yes	no	yes	no
OP_ZEROED	OPERANDS_ARE_ZEROED: Addresses reset	BOOL	yes	no	yes	no
IL_ENABLED	INTERLOCK_IS_ENABLED: Interlocks not deactivated	BOOL	yes	no	yes	no
SV_ENABLED	SUPERVISION_IS_ENABLED: Supervisions not deactivated	BOOL	yes	no	yes	no
ACKREQ_ENABLED	ACKNOWLEDGE_REQUIRED_IS_ENABLED: Mandatory acknowledgment activated	BOOL	yes	no	yes	no

Parameter	Description	Data Type	Read int.	Write int.	Read Ext.	Write Ext.
SSKIP_EN ABLED	STEP_SKIPPING_IS_ENABLED: Skip step activated	BOOL	yes	no	yes	no
SACT_DISP	ACTIVE_STEPS_WERE_DISPLAYED Display list with active steps only	BOOL	yes	no	yes	no
SEF_DISP	STEPS_WITH_ERROR_FAULT_WERE_ DISPLAYED: Display list only with steps with error and disturbed steps	BOOL	yes	no	yes	no
SALL_DISP	ALL_STEPS_WERE_DISPLAYED: Display list with all steps	BOOL	yes	no	yes	no
AUTO_ON	AUTOMATIC_IS_ON: Indicates the automatic mode	BOOL	yes	no	yes	no
TAP_ON	T_AND_PUSH_IS_ON: Indicates the inching mode	BOOL	yes	no	yes	no
TOP_ON	T_OR_PUSH_IS_ON: Display SW_TOP mode	BOOL	yes	no	yes	no
MAN_ON	MANUAL_IS_ON: Indicates the manual mode	BOOL	yes	no	yes	no

15.10 The Transition Structure

A structure with the following parameters is created in the instance DB for every transition located in the sequencer of the S7-GRAPH FB:

Compo-nent	Description	Data Type	Read int.	Write int.	Read Ext.	Write Ext.
TV	Transition is valid	BOOL	yes	no	yes	no
TT	Transition is satisfied	BOOL	yes	no	yes	no
TS	Transition switches	BOOL	yes	no	yes	no
CF_IV	The CRIT_FLT entry is invalid	BOOL	yes	no	yes	no
TNO	Indicates the user-defined transition number	INT	no	no	yes	no
CRIT	Status of the maximum 32 LAD/FBD elements of the transition in the current processing cycle	DWORD	yes	no	yes	no
CRIT_OLD	Status of the maximum 32 LAD/FBD elements of the transition in the previous processing cycle	DWORD	yes	no	yes	no
CRIT_FLT	Copy of CRIT if an error occurs	DWORD	yes	no	yes	no

"Memory minimized" Option

If you set the "Memory minimized" option, an array with the structures of the transitions is created analogous to the "Structure fields" option. Only the Boolean components TV, TT, TS are stored as structure elements. The internal and external accesses are also the same as with the "Structure fields" option.

"Structure arrays" Option

If you have selected the "Structure arrays" option, an array containing the structures of the transitions is created.

Array Name	Length in Bytes
G7T	(number of transitions) ∗ 16

Internal access uses the transition number followed by the name of the structural element.

This means that the following access is possible in conditions or actions:

- T003.TT
- T017.CRIT

External access uses only the array element followed by the name of the structural element.

Caution: The array index is the internal transition number and not the number assigned by the user.

- G7T[1].TT
- G7T[3].CRIT

"Individual structures" Option

If you have selected the "Individual Structures" option, the structures of the transitions are entered as separate sections in the instance DB, one after the other as follows:

Structure Name	Length in Bytes
<Aktionen_Transition>	16
...	
<ASCII_Transition>	16

Internal access uses the transition number or the transition name followed by the name of the structural element.

This means that the following access is possible in conditions or actions:

- T003.TT
- T017.CRIT
- VALVE_ON.TT

External access uses only the transition name (structure name) followed by the name of the structural element.

- VALVE_ON.TT
- VALVE_ON.CRIT

15.11 The Step Structure

A structure with the following components is created in the instance DB for every step of the sequencer in the S7-GRAPH FB, as follows:

Component	Description	Data Type	Read int.	Write int.	Read Ext.	Write Ext.
S1	Step is activated	BOOL	yes	no	yes	no
L1	interlock leaving state	BOOL	yes	no	yes	no
V1	Supervision entering state	BOOL	yes	no	yes	no
R1	Reserved	BOOL	no	no	no	no
A1	Error is acknowledged	BOOL	yes	no	yes	no
S0	Step is deactivated	BOOL	yes	no	yes	no
L0	Interlock entering state	BOOL	yes	no	yes	no
V0	Supervision leaving state	BOOL	yes	no	yes	no
X	Step is active	BOOL	yes	no	yes	no
LA	Interlock is not satisfied	BOOL	yes	no	yes	no
VA	Supervision active	BOOL	yes	no	yes	no
RA	Reserved	BOOL	no	no	no	no
AA	Reserved	BOOL	no	no	no	no
SS	System-internal	BOOL	no	no	no	no
LS	Direct result of the programmed interlock	BOOL	yes	no	yes	no
VS	Direct result of the programmed supervision	BOOL	yes	no	yes	no
SNO	User step number	INT	no	no	yes	no
T	Total step activation time	TIME	yes	no	yes	no
U	Step activation time without disturbance	TIME	yes	no	yes	no
CRIT_LOC	Status of the maximum 32 LAD/FBD elements in the interlock in the current processing cycle	DWORD	yes	no	yes	no
CRIT_LOC_ ERR	Copy of CRIT_LOC when the interlock leaves the state	DWORD	yes	no	yes	no
CRIT_SUP	Status of the maximum 32 LAD/FBD elements in the supervision in the current processing cycle.	DWORD	yes	no	yes	no
SM	System-internal	BOOL	no	no	no	no
LP	System-internal	BOOL	no	no	no	no
LN	System-internal	BOOL	no	no	no	no
VP	System-internal	BOOL	no	no	no	no
VN	System-internal	BOOL	no	no	no	no
H_IL_ERR	System-internal	BYTE	no	no	no	no
H_SV_FLT	System-internal	BYTE	no	no	no	no
RESERVED	Reserve	DWORD	no	no	no	no

"Structure arrays" Option

If you selected the "Structure arrays" option, a field with the structures of the steps is generated, as follows:

Array Name	Length in Bytes
G7S	(number of steps) * 32

Internal access uses the step number followed by the name of the structural element.

This means that the following access is possible in conditions or actions:

- S003.X
- S017.T

External access uses only the array element followed by the name of the structural element.

Caution: The array index is the internal step number and not the number assigned by the user.

- G7S[1].X
- G7S[3].T

"Memory minimized" Option

If you set the "Memory minimized" option, an array with the structures of the steps is created analogous to the "Structure fields" option. Only the Boolean components and the step activation time T are stored as structure elements.

The internal and external accesses are also the same as with the "Structure fields" option.

"Individual structures" Option

If you selected the "Individual Structures" option, the structures of the steps are entered in the instance DB as separate sections, as follows:

Structure Name	Length in Bytes
<Step1>	32
...	
<Stepn>	32

Internal access uses the step number or the step name followed by the name of the structural element.

This means that the following access is possible in conditions or actions:

- S003.X
- S017.T
- DRY.X

External access uses only the step name (structure name) followed by the name of the structural element, as follows:

- DRY.X
- DRY.T

15.12 Sequencer Status

The sequencer status area consists of several components (arrays or structures) containing information about the status of the sequencer, as follows:

- Permanent Instructions Component
- Index Arrays Component
- Mode Component (Mode of Operation, MOP)
- Activation Time Component
- Group Bits Component

The sequencer status area starts at the address: a = FB parameter length + transition array length(s)+step array length(s).

Permanent Instructions

This component of the sequencer status area describes the permanent instructions that can be programmed before or after the sequencer. The component is omitted if no permanent instructions are programmed.

Name	Description	Data type	Length	Read int.	Write int.	Read Ext.	Write Ext.
PRE_CRIT	Status of the permanent instruction preceding the sequencer	DWORD	0 - n	no	no	yes	no
POST_CRIT	Status of the permanent instructions following the sequencer	DWORD	0 - n	no	no	yes	no
PRE_CNT	Number of permanent instructions preceding the sequencer	WORD		no	no	yes	no
POST_CNT	Number of permanent instructions after the sequencer	WORD		no	no	yes	no

Index Arrays

The first component of the sequencer status area contains the index arrays for managing the internal step and transition numbers.

Index Name	Description	Data Type	Length in Bytes	Read int.	Write int.	Read Ext.	Write Ext.
TVX	Array of valid transitions	BYTE	<intern>	no	no	yes	no
TTX	Array of satisfied transitions	BYTE	<intern>	no	no	yes	no
TSX	Array of switching transitions	BYTE	<intern>	no	no	yes	no
S00X	Array of steps to be deactivated (second last cycle)	BYTE	<intern>	no	no	no	no
SOFFX	Array_1 of the steps to be deactivated	BYTE	<intern>	no	no	no	no
SONX	Array_1 of the steps to be activated	BYTE	<intern>	no	no	no	no
SAX	Array of active steps	BYTE	<intern>	no	no	yes	no
SERRX	Array of disturbed steps	BYTE	S_CNT+2	no	no	yes	no
SMX	Array of marked steps	BYTE	S_CNT+2	no	no	no	no
S0X	Array_2 of the steps to be deactivated	BYTE	S_CNT+2	no	no	no	no
S1X	Array_2 of the steps to be activated	BYTE	S_CNT+2	no	no	no	no
S_DISPLAY	Copy of the output parameter S_NO (can also be evaluated if the minimum parameter set option is selected)	INT	2	yes	no	yes	no
S_SEL_OLD	System-internal	INT	2	no	no	no	no
S_DISPIDX	System-internal	BYTE	1	no	no	no	no
T_DISPIDX	System-internal	BYTE	1	no	no	no	no

Length of the Index Arrays

The length of the index arrays TVX to S1X is obtained as follows:

l = (maximum number of elements) + 2

The index arrays are structured as follows:

Array[0]	=	Index of the first free position in the array
Array[1] - Array[n-2]	=	Entries
Array[n-1]	=	0 (indicates the end of the array)

New entries are added at the end. When entries are removed, the array is shifted together. This means that, for example, in the array of active steps (SAX), the step that is active longest has the lowest index.

Modes

The next component in the sequencer status area describes the modes, in other words the state or mode change and codes.

A structure with the following elements is created:

Name	Description	Data Type	Read int.	Write int.	Read Ext.	Write Ext.
AUTO	Change to the automatic mode	BOOL	yes	yes	yes	yes
MAN	Change to the manual mode	BOOL	yes	yes	yes	yes
TAP	Change to the inching mode	BOOL	yes	yes	yes	yes
TOP	Change to the automatic or switch to next mode	BOOL	yes	yes	yes	yes
ACK_S	Reserved	BOOL	no	no	no	no
REG_S	Reserved	BOOL	no	no	no	no
T_PREV	Reserved	BOOL	no	no	no	no
T_NEXT	Reserved	BOOL	no	no	no	no
LOCK	Activate/deactivate interlock processing	BOOL	yes	yes	yes	yes
SUP	Activate/deactivate supervision processing	BOOL	yes	yes	yes	yes
ACKREQ	Reserved	BOOL	no	no	no	no
SSKIP	Reserved	BOOL	no	no	no	no
OFF	Deactivate sequencer	BOOL	yes	yes	yes	yes
INIT	Activate initial steps	BOOL	yes	yes	yes	yes
HALT	Stop/continue sequencer	BOOL	yes	yes	yes	yes
TMS_HALT	Stop/continue timers	BOOL	yes	yes	yes	yes
OPS_ZERO	Deactivate/activate actions	BOOL	yes	yes	yes	yes
SACT_DISP	Reserved	BOOL	no	no	no	no
SEF_DISP	Reserved	BOOL	no	no	no	no
SALL_DISP	Reserved	BOOL	no	no	no	no
S_PREV	Automatic mode: Indicate previous simultaneously active step in S_NO Manual mode: Indicate next lower step number	BOOL	yes	yes	yes	yes
S_NEXT	Automatic mode: Indicate next simultaneously active step in S_NO Manual mode: Indicate next higher step number	BOOL	yes	yes	yes	yes
S_SELOK	Reserved	BOOL	no	no	no	no
S_ON	Manual mode: Activate the displayed step	BOOL	yes	yes	yes	yes
S_OFF	Manual mode: Deactivate the displayed step	BOOL	yes	yes	yes	yes
T_PUSH	Mode: Inching Transition passes control when the satisfied condition is accepted with T_PUSH	BOOL	yes	yes	yes	yes
REG	Reserved	BOOL	no	no	no	no

Name	Description	Data Type	Read int.	Write int.	Read Ext.	Write Ext.
ACK	Acknowledgment of a disturbance	BOOL	yes	yes	yes	yes
IL_PERM	Permanent interlock processing	BOOL	yes	yes	yes	yes
T_PERM	Permanent transition processing	BOOL	yes	yes	yes	yes
ILP_MAN	Permanent interlock processing in manual mode	BOOL	yes	yes	yes	yes

Indicating the Modes

The modes component is saved in the instance DB as follows:

Structure Name	Description	Length in Bytes
MOPEDGE	System-internal	4
MOP	Changing modes and indicators	4

Example:

By setting MOP.MAN once, the mode changes to MANUAL. This can be set either controlled by the program (for example action in the sequencer) or by the PG function "Modify Variable" or using an operator panel.

Activation Time

The following component consists of a structure containing information about the activation time calculation. This is only evaluated internally by the system.

Structure Name	Description	Length in Bytes
TICKS	System-internal	12

Group Bits

The last component of a sequencer status area contains group bits that monitor the internal sequencer status. A structure is created as follows for these group bits:

Name	Description	Data Type	Read int.	Write Int.	Read Ext.	Write Ext.
ERR_FLT	Group disturbance	BOOL	yes	no	yes	no
ERROR	interlock error	BOOL	yes	no	yes	no
FAULT	Supervision error	BOOL	yes	no	yes	no
RT_FAIL	Group indication: Runtime error	BOOL	yes	no	yes	no
NO_SNO	Selected step does not exist	BOOL	yes	no	yes	no
NF_OFL	Too many steps to activate or deactivate	BOOL	yes	no	yes	no
SA_OFL	To many active steps	BOOL	yes	no	yes	no
TV_OFL	Too many valid transitions	BOOL	yes	no	yes	no
MSG_OFL	Too many messages	BOOL	yes	no	yes	no
NO_SWI	Reserved	BOOL	no	no	no	no
CYC_OP	System-internal	BOOL	no	no	no	no
AS_MSG	System-internal	BOOL	no	no	no	no
AS_SEND	System-internal	BOOL	no	no	no	no
SQ_BUSY	System-internal	BOOL	no	no	no	no
SA_BUSY	System-internal	BOOL	no	no	no	no
AS_SIG	System-internal	BOOL	no	no	no	no

The component with group bits is saved in the instance DB as follows:

Structure Name	Type	Description	Length in Bytes
SQ_FLAGS	t_sq_flags	Group bits	2

15.13 Internal Data Area

This part of the instance DB is a system-internal area. These data should not be accessed. The data area is divided into 4 sections:

Internal Working Area 1

Description	Data	Length
Internal Working Area 1	Variable	Variable

Condition Descriptions

These data are only entered when you select the option "With Data".

Address	Description	Data	Length
	ADR_LOG_T[]: Descriptions for all transitions with content; the start addresses are in the field ADR_LOG_T[]; a 0 is entered here for empty transitions and there is no description. LOG_Ti[] LOG_Tj[] ... LOG_Tk[] This is possibly followed by padding bytes with the value 0 at the end of LOG_Tx so that the following fields start at an even address.	BYTE[] BYTE[] ... BYTE[]	
	ADR_LOG_L[]: Descriptions of all interlock conditions with contents; the start addresses are in the field ADR_LOG_L[]; here, a 0 is entered for empty interlocks and there is no description. LOG_Lp[] LOG_Lq[] ... LOG_Lr[] This is possibly followed by padding bytes with the value 0 at the end of LOG_Ly, so that the following fields start with an even address	BYTE[] BYTE[] ... BYTE[]	
c∗ S_CNT-2∗ T_CNT	**ADR_LOG_T:** Addresses of the transition descriptions LOG_Tx ADR_LOG_T[0...T_CNT]	WORD[]	(T_CNT+1)∗2
c+2∗ T_CNT+2	**ADR_LOG_L:** Addresses of the interlocks LOG_Ly ADR_LOG_L[0...S_CNT]	WORD[]	(S_CNT+1)∗2

Address	Description	Data	Length
c+2∗ T_CNT+2∗ S_CNT+4	**ADR_T1_CRITFLT:** Address of disturbed transition criteria of T1 ADR_T1_CRITFLT == address(G7T[1].CRIT_FLT)	WORD	2
c+2∗ T_CNT+2∗ S_CNT+6	**ADR_S1_CRITERR:** Address of the disturbed interlock criteria of S1 ADR_S1_CRITERR == address(G7S[1].CRIT_ERR)	WORD	2
c+2∗ T_CNT+2∗ S_CNT+8	**ADR_SERRX:** Address of the array of disturbed steps SERRX[] ADR_SERRX == address(SERRX[0]) == e	WORD	2

n = Length of the instance DB in bytes

Internal Work Area 2

Description	Data	Length
Internal Work Area 2	Variable	Variable

Numeric Data of the Sequencer

Address	Description	Data	Length
n-36	**ADR_COND:** Address for the end of the condition descriptions ADR_COND == Address(ADR_LOG_T[0]) == c	WORD	2
n-34	**MAX_TVAL:** Max. number of simultaneously valid transitions	BYTE	1
n-33	**MAX_SACT:** Max. number of simultaneously active steps	BYTE	1
n-32	**MAX_CRIT:** Max. number of criteria per condition	BYTE	1
n-31	**T_CNT** Number of transitions	BYTE	1
n-30	**SUP_CNT:** Number of supervisions	BYTE	1
n-29	**LOCK_CNT:** Number of interlocks	BYTE	1
n-28	**S_CNT** Number of steps	BYTE	1
n-27	**SQ_CNT:** Number of branch paths	BYTE	1
n-26	**PRS_INPAR:** Range of input parameters	DWORD	1
n-22	**PRS_OUTPAR:** Range of output parameters	DWORD	1

Address	Description	Data	Length
n-18	**SQ_FLAGS** Compiler flags part 1	WORD	6
n-6	**SQ_FLAGS** Compiler flags part 2 Bit .0 == 1 --> condition descriptions exist	BYTE	1
n-5	**SQ_FLAGS** Compiler flags part 3	BYTE	1
n-4	**G7_FBMARK:** Compatibility FB criteria analysis (010 == 1.0)	BYTE	1
n-3	**G7_RELEASE:** Graph version number (010 == 1.0)	BYTE	1
n-2	**G7_ID2:** Graph ID ("7")	BYTE	1
n-1	**G7_ID1:** Graph ID ("G")	BYTE	1

n = Length of the instance DB in bytes

Glossary

--A--

Action

An action is an instruction to execute a function in an active step.

--C--

Checkpoint

A checkpoint is a point in the program cycle at which plant or process states can be influenced or queried. There are various checkpoints, for example

- System checkpoint
- User checkpoint

Criteria analysis

Criteria analysis is a software program for S7 Graph, consisting of a criteria FB and extended instance DB. Criteria analysis detects and influences plant and process states of S7 Graph sequential control systems.

--D--

Deactivate

Deactivating means that the step is terminated by the relevant transition or by another step.

--E--

Edge evaluation

Edge evaluation is required to detect and evaluate the change in a signal state.

The term edge is used when a signal state (for example of an input) changes. A **rising edge** means that the signal changes from state "0" to state "1". In the opposite situation, there is a **falling edge**.

Execution time

The execution time of an instruction is the time during which the instruction is active; in other words, executed.

--I--

Interlock

An interlock is a programmed condition within a step. An interlock influences the execution of individual actions.

--L--

LAD branch

The LAD branch corresponds to an OR logic operation on elements in a transition, interlock, or supervision.

--N--

Normally-open contact/normally-closed contact

Normally-open contacts and normally-closed contacts are bit logic operations. They can be used as elements in transitions, supervisions, or interlocks.

- The normally-open contact switches when the Boolean address is **set**.
- The normally-closed contact switches when the Boolean address is **not set**.

--P--

Path in an alternative branch

A path in an alternative branch is the divergence of a sequencer to a specific step dependent on several parallel transitions.

Process image

A memory area on the CPU in which the status of the inputs and outputs is entered.

1. Process image input table (PII): PII
2. Process image output table (PIQ): PIQ.

Processing time

The processing time is the time taken by the CPU to process an instruction (action, condition) and is usually in the ms range per instruction.

--S--

Sequencers

A sequencer is the heart of a sequential control system. Here, the program for sequential control is created and processed.

The sequencer consists of a series of steps whose execution is started in a specified dependent on step enabling conditions.

Sequential control system

A control system in which control is passed sequentially from one step to the next according to the program dependent on conditions.

Step

The control task is divided into single steps. Actions are formulated in the steps that are executed by the controller in a certain status (for example for controlling the outputs or activating and deactivating steps).

Step comment

The step comment is a text in a step with a maximum of 2048 characters.

Start point

The start point is the defined start of the program cycle. When the program starts, the process image input table (PII) is read, effectively querying the plant or process state.

Status

The status is the signal state of the bits of an address on the PLC.

Status display

The status display is the display of the signal state of an address on the monitor of a programming device/PC connected online with the PLC.

Step enabling condition

Step enabling conditions are Boolean logic operations that specify the conditions for terminating the predecessor step and activating the successor step.

They are components of the transitions.

Structural element

Structural elements are jumps and transitions in a sequencer.

Supervision

A supervision is a programmable condition within a step. A supervision influences the way in which control passes from one step to the next.

System checkpoint

The system checkpoint is the interface between the system and the user program. At the system control point, you can monitor or modify the signal states of the process variables and set the signal states at outputs from the programming device.

--T--

Transition

A transition is the part of the sequencer that contains the conditions for passing control from one step to its successor step.

A valid transition is a transition whose predecessor step or steps are active.

When a transition is valid and its step enabling conditions are satisfied, control passes to the successor step. The predecessor step is then deactivated and the successor step is activated.

--U--

User checkpoint

The user checkpoint is a selectable point in the user program for displaying the signal states of variables.

User program

The user program contains all the blocks and instructions of the program.

Index

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

Z