SIEMENS

SIMATIC

DDE Server Configuration and Administration

User Manual

C79000–G7076–C807–01

Version 2

Safety Guidelines

This manual contains notices which you should observe to ensure your own personal safety, as well as to protect the product and connected equipment. These notices are highlighted in the manual by a warning triangle and are marked as follows according to the level of danger:

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 or property damage can result if proper precautions are not taken.

Note

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

Qualified Personnel

The device/system may only be set up and operated in conjunction with this manual.

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® and SINEC® 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
Automation Group
Industrial Automation Systems
Postfach 4848, D-90327 Nürnberg

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.

Technical data subject to change.

Siemens Aktiengesellschaft

Order No. C79000–G7076–C807–01

Preface

Purpose

The purpose of this manual is:

- To explain the basic concepts of the SIMATIC DDE Server
- To introduce its functions

The SIMATIC DDE Server

The SIMATIC DDE Server is used universally to link automation components to MS Windows applications. This means that a wide range of standard MS Windows application programs is made available for integration into automation solutions.

Audience

This manual is intended for persons who wish to use the DDE Server to link automation components to MS Windows applications.

Experience

A basic knowledge of MS Windows, DDE application programming and SIMATIC S5 and S7 Programmable Logic Controllers is assumed.

Scope

This manual is valid for the following software products:

- SIMATIC PC DDE Server Windows 3.11 Version 2.0
- SIMATIC M7 DDE Server Windows 3.11 Version 2.0
- SIMATIC M7 DDE Server Windows 95 Version 2.0

The name SIMATIC DDE Server is used in the universally valid document parts. The complete product name e.g. SIMATIC PC DDE Server Windows 3.11 is used only in descriptions related exclusively to this product.

Overview of Contents

This manual discusses the concept, installation, and use of the SIMATIC DDE Server. Linking to MS Excel applications is illustrated by means of examples.

- Overview of product and DDE concept (Chapters 1, 2, 5 and 6)
- Installation (Chapters 3 to 4)
- Using the DDE Server via the interactive interface (Chapters 7 to 8)
- Commands (Chapter 9)
- Examples of conversation with MS Excel (Chapter 10).
- Reference information (Appendix)

Not included in this Manual...

You will find further information on how to use the SIMATIC DDE Server and on error messages in the online help.

A detailed description of the DDE mechanism and the general DDE interface will be found in the corresponding MS Windows publications.

References to other manuals are shown using the part number of the literature between slashes /.../. Using these numbers you can find out the exact title of the manual from the literature list at the end of this manual.

Contents

Product Overview 1

Overview

By using the SIMATIC DDE Server, it is possible to integrate MS Windows applications into an automation solution. The DDE access mechanism featured by Windows allows online reading from and writing to process variables.

In this Chapter

1.1 Application

Automation with Windows

Made possible by the DDE Server, integration of automation components into the MS Windows environment presents you with the user interface with which you are familiar. It puts you in a position to integrate standard applications – for example, MS Excel – into automation solutions and thus drastically to cut the cost of software production.

MS Windows standard applications

A wide variety of software packages based on the Windows 3.11 and Windows 95 operating systems are available for PCs. The products on offer include tools for word processing, statistics, spreadsheets, database applications, quality assurance, process visualization, and production automation. Most standard programs use a macro language in order to exchange data with other programs by supporting the DDE mechanism.

User – own MS Windows applications

You can even integrate your own MS Windows applications that have been implemented in any programming language into automation solutions.

Use the Windows DDEML application programming interface to implement applications capable of DDE communication. See /**4**/ for descriptions of the DDEML function calls.

1.2 Product Structure

Products described

This manual describes the following products:

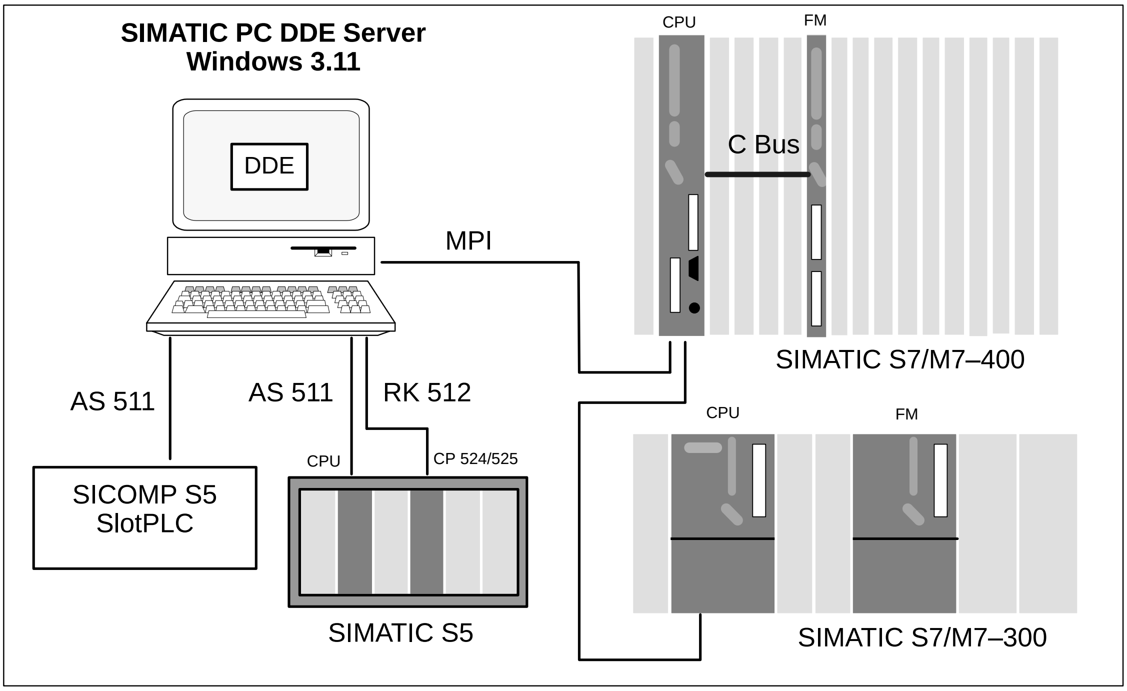
Product	Order No.
SIMATIC PC DDE Server Windows 3.11 Version 2.0	6ES7807 – 7AA00 – 0YX0
SIMATIC M7 DDE Server Windows 3.11 Version 2.0	6ES7807 – 2AA00 – 0YX0
SIMATIC M7 DDE Server Windows 95 Version 2.0	6ES7807 – 2BA00 – 0YX0

These products differ by the hardware platform and by the operating system used. They also offer different interfaces to access the automation systems.

SIMATIC PC DDE Server Windows 3.11

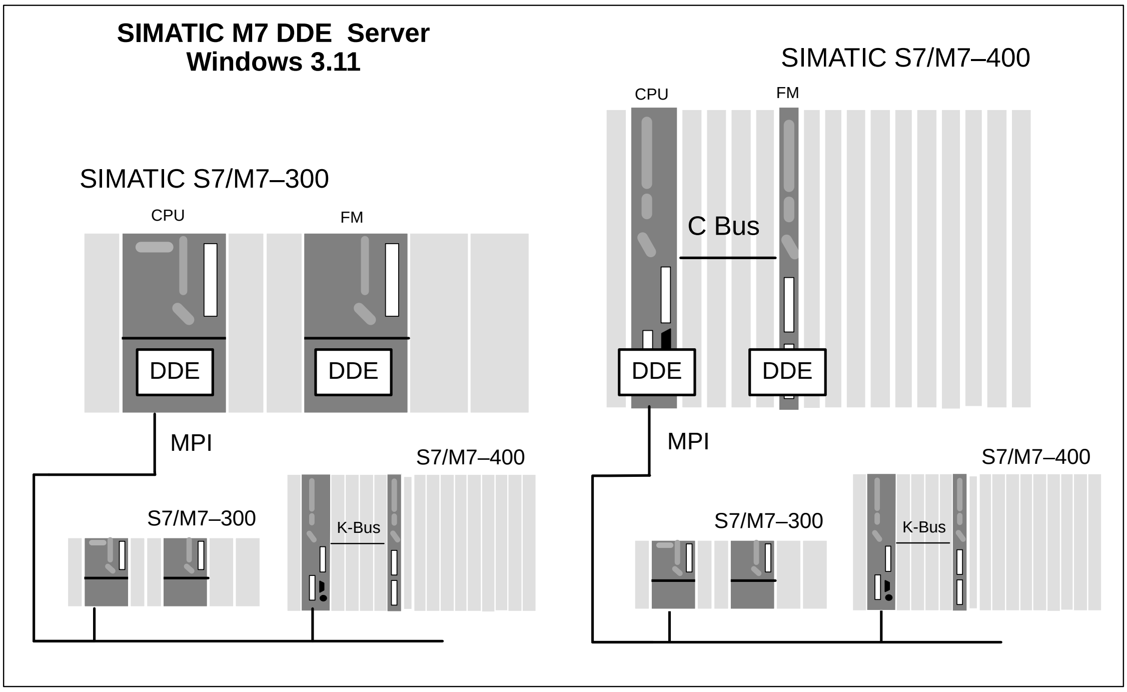
This DDE Server runs on a PC with Windows 3.11. It can access up to 16 automation systems of the following types

- SIMATIC S5 automation systems with serial interface via the AS 511 and RK 512 protocols.
- SICOMP S5 Slot PLC (controller co – processor for industrial PCs having the functionality of a PLC) via the AS 511 protocol.
- SIMATIC S7/M7 automation systems via the Multipoint Interface (MPI)



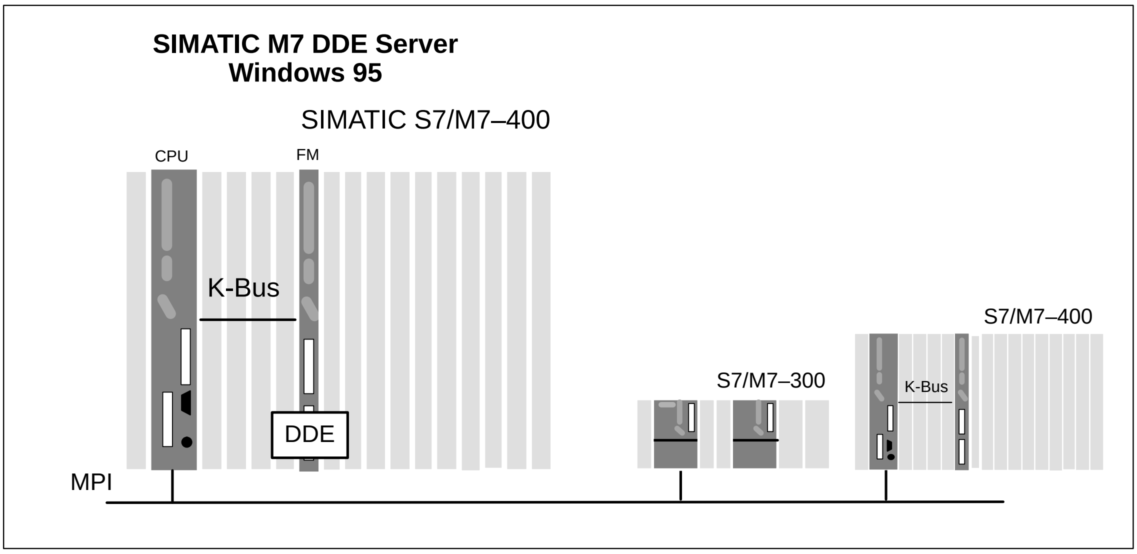
SIMATIC M7 DDE Server Windows 3.11

This DDE Server runs on a SIMATIC M7 CPU or FM with Windows 3.11. It can access up to 16 SIMATIC S7/M7 automation systems via via the Multipoint Interface (MPI) and the communication bus (K bus).



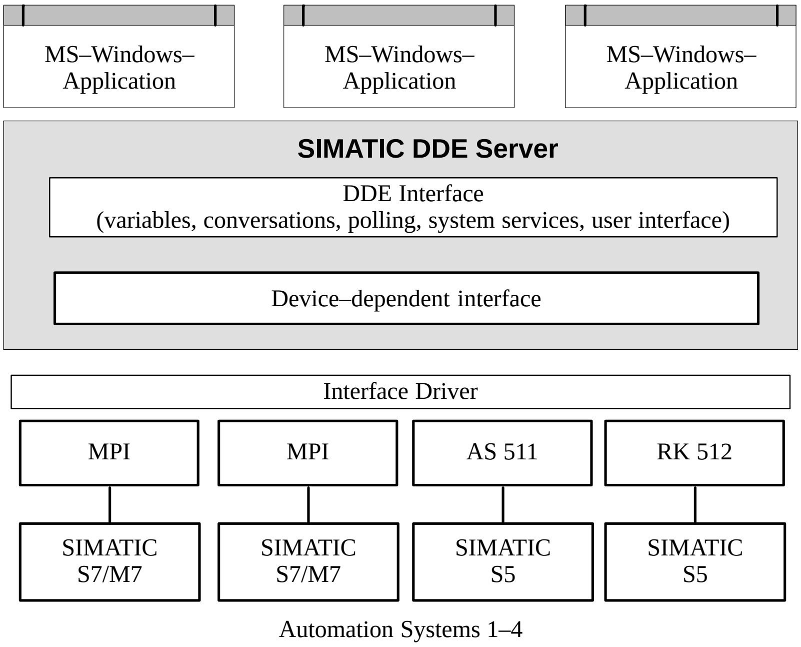
SIMATIC M7 DDE Server Windows 95

This DDE Server runs on a SIMATIC M7 FM 456 – 4 with Windows 95. It can access up to 16 SIMATIC S7/M7 automation systems via via the Multipoint Interface (MPI) and the communication bus (K bus).



Structure of DDE Server

The structure of the DDE Server is the same in all three products. Only the provided interface drivers and hence the possibly accessed automation systems are different. The following figure shows as an e×ample the structure of the SIMATIC PC DDE Server Windows 3.11.



Application

The MS Windows application accesses the Server via the DDE interface defined in MS Windows.

DDE Server

To be able to implement the link to a new automation system with utmost flexibility, the DDE Server is divided into two parts:

- DDE interface and general services

  This module includes the DDE commands, the user interface, general configuration, and variable and connection management.

- Device – dependent interface

  For every device type, the device – dependent interface consists of a driver DLL used to adapt the automation system to the DDE interface.

Interface Driver

The interface driver establishes the link to the automation system.

Separate interface drivers are not required if this part has already been implemented in the device – dependent interface.

1.3 Functionality

All three types of SIMATIC DDE Server provide the same functionality, i.e. communication services and administrative services.

Communication services

The communication services within the meaning of DDE include:

- reading and writing of variables to automation systems and
- execution of commands.

Administrative services

The administrative services include:

- Server configuration
- Trace functionality for test purposes
- monitoring and statistics functions.

Standard user interface

The DDE Server provides its services via the MS Windows standard graphical user interface for PCs. The DDE Server thus opens up the Windows user interface for SIMATIC M7 and PC–based automation solutions.

DDE Concept

2

Overview

DDE (dynamic data exchange) is a method of communication provided by MS Windows for exchanging data between two applications. This mechanism is supported by nearly all MS Windows – based applications – for example, MS Excel, MS Word, Visual Basic, Object Vision, etc.

In DDE a protocol is defined. The basis for the DDE protocol is the Windows – specific messaging system. Two Windows – based applications engage in a DDE conversation by exchanging messages and communicating with each other according to this defined communication protocol.

This chapter provides the reader with an introduction to the DDE mechanism and a basic explanation of the addressing of data when applications use the DDE mechanism to talk to each other.

In this Chapter

2.1 Client/Server Principle

A DDE conversation is based on the client/server principle.

Client

The application that initiates a DDE conversation and controls the exchange of data is the **client**. The client controls the logic communication channel, opens it, sends requests, and closes it again.

Server

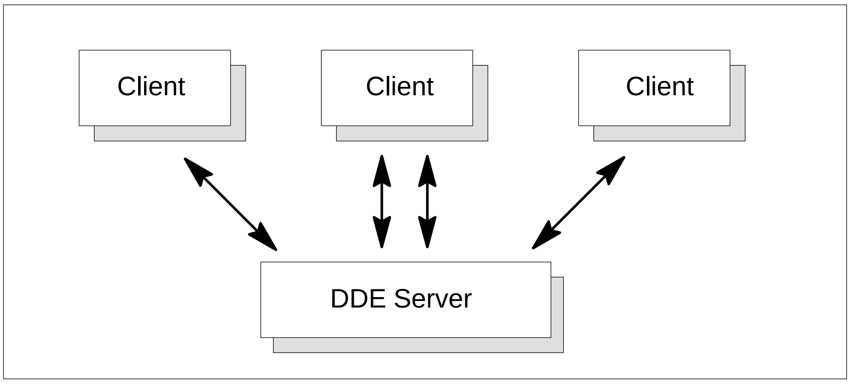
The application that responds to the requests for information from the client is the **server**. The server is the passive partner in the conversation and responds only to requests for information from the active partner, i.e. the client. From the viewpoint of DDE, the SIMATIC DDE Server is always a server that provides services for a MS Windows application (client) when requested to do so.

Data can be exchanged on a one – time or on a continuous basis.

Possible connections

Several clients can open connections to the SIMATIC DDE Server.

A client can open more than one connection to the Server. The client can access several variables per connection.



A direct conversation between two clients or two servers is not possible.

2.2 DDE Conversation

Conversation procedure

A typical sequence of events in a DDE conversation is as follows:

1. The DDE client initiates a DDE conversation by opening a connection to the Server. The DDE Server acknowledges opening of the DDE connection.
2. The applications exchange data.
3. The client or server terminates the DDE conversation.

Types of Link

There are three different types of link: **cold link**, **warm link**, and **hot link**.

Cold link

One – Time reception of data following a one – time request from the client or one – time data transfer by the client to the Server

Receive:

The client sends a request for data to the Server.
If the Server is in a position to provide these data, it sends them to the client. If not, it sends a negative acknowledgment.

Requests by the client for information can be repeated as often as required for other items until the client or Server closes the DDE connection.

A cold – link conversation is used to read variables. In the case of dynamically changing data in an automation system, the client can request the same variable time and time again to stay up to date. In this particular instance, depending on the frequency of requests for information, there is no guarantee that all changes to the variable will be detected. On the other hand, the majority of requests for information would remain ineffective if data were to change only slowly. This is the reason why DDE features more efficient types of conversation for dynamically changing data.

Send:

The client sends data to the Server on a one – time basis (poke).

Hot link

Continuous reception of data following a one – time request

With this type of conversation, the client sends a one – time request for data to the Server. This means that the Server is tasked to advise the client whenever the value of a particular item has changed.

When the Server acknowledges this request, it sends all changes to the values of that item thereafter to the client without the client having to issue another request to the Server.

Changes are sent until the client terminates data connection.

This type of conversation is suitable for requesting changes to data and is used when the client wishes to update a variable. As soon as updated data are available, they are sent over the DDE link. This ensures that even sporadically changing data are processed in the Server.

Warm link

Continuous reception of messages (as opposed to data) following a one – time request from the client

This type of conversation is a combination of cold and hot links. Unlike the hot link, however, the client does not receive updated data without first requesting them. Its attention is simply drawn to the fact that data have changed. It is then up to the client to decide if and when it fetches them from the Server.

This type of conversation is used when the client wishes to be informed about an updated variable but wants to control data reception itself.

2.3 Addressing Data in DDE Conversations

Addressing

To request data from a Server, the client uses an address specification consisting of three components:
service, **topic**, and **item**

From the specification made by the client, the Server determines the data and transfers them to client in a DDE conversation.

Service

The **service** component refers to the **name of the DDE Server** in the Windows environment.

The SIMATIC DDE Server is always addressed as **ddeplc**.

Topic

The **topic** component limits in the Server an area for those data which the client wishes to access. The topic depends on the Server application.

With the SIMATIC DDE Server the two standard topics **default** and **system** are supported.

- default: With this topic the automation systems are addressed.
- system: With this topic, general information about the DDE interface of the SIMATIC DDE Server is transferred.

Note

The transfer of a blank string as a **topic** or a blank string as a **service** in opening a DDE connection is responded to by the SIMATIC DDE Server with positive acknowledgments to each of its topics. It is now the duty of the DDE client concerned to react to these acknowledgments and to close all unselected DDE connections.

Item

The **item** component is a **data item** that is defined in the Server. By means of access to this data item, item values can be transferred from the client to the Server and vice versa. The SIMATIC DDE Server supports the following standard types of items (for more information see Appendix B:

- **System items**

  With the **system** topic, the SIMATIC DDE Server supports items for requesting information about the DDE Servers internal states.

- Non – system items

  With the **default** topic, the SIMATIC DDE Server supports items for requesting information on data formats and items. These items build a subset of the system items.

  Also with the **default** topic the SIMATIC DDE Server supports the access to user data, i.e. to variables of the automation and the execution of commands and options (see Chapter 9).

2.4 Addressing in the SIMATIC DDE Server

Introduction

This section describes the specific aspects of addressing in the SIMATIC DDE Server. All the variables of automation systems are addressed with the **default** topic. Commands and options can be executed with this topic as well.

Variables

Variables correspond to SIMATIC S5/S7 address areas (data blocks, counters, flags, etc.). You can access variables either by

- freely selectable names if you have defined them in a symbol file (see Symbolic Addressing; Chapter 5), or
- their absolute address (See Absolute Addressing; Chapter 6).

Commands

The SIMATIC DDE Server supports the execution of commands, e.g. for loading a symbol file or for declaring a subsystem. Commands are always transferred at the **item** address component. The commands are described in Chapter 9.

Options

Options are used to set certain properties of the DDE conversation. They are transferred in the **topic** address component with the **default** topic.

Symbolic addressing

Symbolic addressing renders your applications independent of address structures and memory locations.

In a symbol file evaluated by the DDE Server, you assign freely selectable variable names and data types to addresses you wish to access. Applications using only these variable names as components in DDE addresses do not have to be adapted by the user if their absolute address (e.g. memory location) changes. Only the symbol file needs modification.

Note

The symbol file is not the symbol table known in STEP 7. It is an ASCII file that contains the SIMATIC DDE Server’s own symbol information.

Absolute addressing

Clients can send the complete address of the requested data to the DDE Server. This means that you can also access data for which variables are not defined in the symbol file. Clients opting for this absolute addressing then depend heavily, however, on the address structure in the Programmable Logic Controller.

Subsystems

Every automation system that is addressed by the SIMATIC DDE Server is referred to as a subsystem. Possible subsystems are:

- SIMATIC S7/M7 automation systems
- SIMATIC S5 Programmable Logic Controllers
- SICOMP S5 SlotPLCs
- test subsystem
  (for testing the DDE Server without a link to a Programmable Logic Controller ; addressed by the simulation driver).

The SIMATIC DDE Server always defines a default subsystem to which the variables have been assigned as default settings.

Subsystems can be declared in the symbol file or by using a command.

Installation and Geting Started 3

Overview

This chapter illustrates how you can quickly test the SIMATIC DDE Server, without Programmable Logic Controllers being connected.

In this chapter

3.1 Installation

Backup copy

To protect your master floppy disks:

1. Make a backup copy of the master floppy disk(s).
2. Always use the backup copy for installation and de–installation.

Installation

Start the installation program **install.exe** from the floppy disk under Windows.

During the installation process you can select the:

- pathname of the directory to which all the files belonging to the DDE Server are to be copied
- language on the user interface (English or German)

Please see also the installation notes contained in the Product Information.

Logbook

The installed components are recorded in the file INSTALL.LOG.

3.2 Starting the DDE Server with the Test Driver

Default Setting

Once installed, the SIMATIC DDE Server is set such that it can be quickly started and tested without having to connect to a specific automation system.

Start the SIMATIC DDE Server using the simulation driver to check the installation and to become accustomed to the user interface and to the way in which the DDE Server operates.

Starting the DDE Server

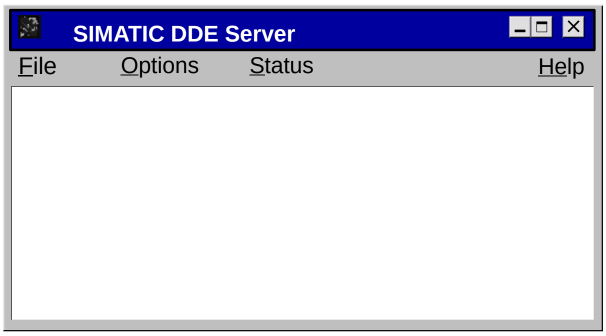
Under Windows 3.11: Activate the icon for the DDE Server in its program group

SIMATIC DDE Server

by double – clicking on it with the left mouse button or by pressing the Return key after selecting the icon.

Under Windows 95: Start the DDE Server with the “Start“ button.

After the startup message, the main window of the DDE Server is displayed:



The test driver provides three variables that change with time. With applications that support DDE, you can read the values of these variables from the DDE Server and thus check it out for correct functioning.

To check how the DDE Server functions, open a connection to an MS Windows application that supports DDE – for example, a MS Excel worksheet.

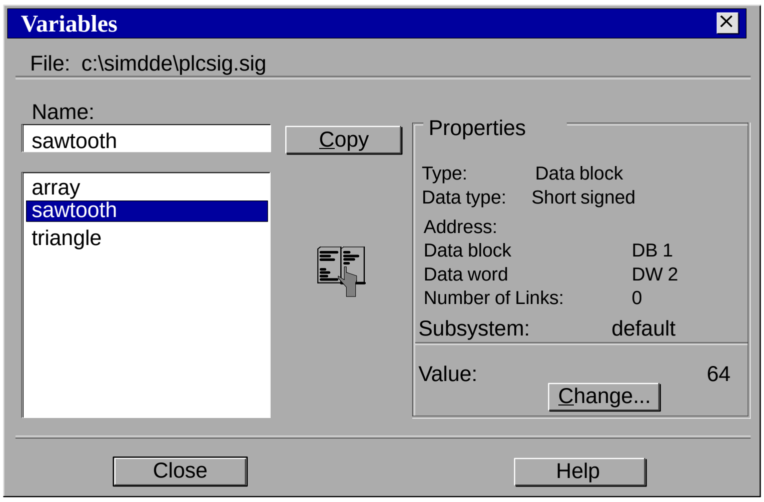
3.3 Displaying a Variable in a MS Excel Worksheet

Procedure

This section shows simple establishment of a DDE conversation with a MS Excel application.

To open a connection to MS Excel:

1. Open the SIMATIC DDE Server window (see Chapter 3.2)
2. Choose **Status ▶ Variables**. The **‘Variables‘** dialog box is displayed.



Variables

File: c:\simdde\plcsig.sig

Name: sawtooth

array
sawtooth
triangle

Properties

Type: Data block
Data type: Short signed
Address:
Data block DB 1
Data word DW 2
Number of Links: 0
Subsystem: default

Value: 64

3. Select one of the three variables from the displayed list.
   The name of the variable then appears in the ‘Name‘ field. The ‘Properties‘ field displays information on the variable you selected.
4. Click on the ‘Copy‘button.
   (The value of the variable is stored on the Windows Clipboard. When the Clipboard is opened, you will see the existing DDE address specification, which consists of three components, for that variable.)
5. Start MS Excel and open a new MS Excel worksheet.
6. Select a cell.
7. Choose, in MS Excel V4 **Edit ▶ Paste Link**
   and in MS Excel V5 **Edit ▶ Insert contents ▶ Link**.
   The current value of the variable you selected appears in the worksheet cell. You can observe how this value varies with time.

This means that you have established a DDE link at a few clicks of the mouse. The value of the variable is immediately updated in the application (MS Excel). Links of this type are known as hot links.

To continue...

Once you have become accustomed to the manner in which the DDE Server functions in simulation mode, you can go on to startup of the Server with an automation system.

System Startup 4

This chapter describes startup for the following automation systems:

- SIMATIC S7/M7 automation systems via MPI
- SIMATIC S5 PLC via serial interface (AS 511)
- SIMATIC S5 PLC via CP computer link board (RK 512)
- SICOMP S5 SlotPLC via serial interface (AS 511)

In this chapter

4.1 Operation with SIMATIC S7/M7 via MPI

What you need to do

To start up the SIMATIC DDE Server with a SIMATIC S7 or SIMATIC M7 automation system, you must take the following action:

1. Set the default subsystem (optional).
2. Create a symbol file if you are going to use symbolic addresses (see Chapter 5)
3. Start the SIMATIC DDE Server and applications.

4.1.1 Requirement

MPI or C Bus Connection

The SIMATIC M7 or the PC must be connected to the SIMATIC S7/M7 automation system either:

- via MPI or
- via communication bus (C bus), if communication takes place within the same SIMATIC station

Setting the MPI Interrupt

For the SIMATIC PC DDE Server Win 3.11 only: The interrupt which is used to address the MPI card must be set with the keyword **HWINT_VECTOR** in the initialization file **s7dpmpi.ini** in the Windows directory under section **[S7MPISPC2]**.

```
[S7MPISPC2]

HWINT_VECTOR=11
```

Caution

If a wrong interrupt number is set in the file s7dpmpi.ini, unexpected system behavior can occur and the system may become inoperable. The effect depends on the device this interrupt was assigned to.

Operating several interfaces

The driver is capable of accessing up to 16 SIMATIC S7/M7 automation systems via MPI or K bus concurrently.

Every automation system forms a subsystem. Subsystems have to be made known to the DDE Server by either

- an entry on the symbol file, or
- the *susbsys* command.

The subsystem entry in a symbol file is described in Appendix A; you will find the *subsys* command in Chapter 9.

4.1.2 Setting the Interface as the Default Subsystem

Procedure

To define the MPI interface as the default subsystem:

1. Start the DDE Server – for example, by activating the appropriate icon on the MS Windows user interface.
2. In the menu bar of the DDE Server main window choose **Options ▸ Configuration...**
   The “Configuration” dialog box is displayed.
3. Click on the drop button in the Drivers field and select the driver **mpi.drv** from the opened drop–down list. The name of the driver is then visible in the viewing section.
4. Click on the OK button.
5. If you have changed the default DLL, exit from the DDE Server by choosing **File ▸ Exit** and restart it.

4.1.3 Starting the SIMATIC DDE Server and Application

1. Start the SIMATIC DDE Server.
2. Start your application(s).

4.2 Operation with SIMATIC S5 PLC via Serial Interface AS 511 (for SIMATIC PC DDE Server Windows 3.11 only)

What you need to do

To start up the SIMATIC PC DDE Server Windows 3.11 with a SIMATIC S5 automation system, you must take the following action:

1. Install the hardware driver for communication with the automation system.
2. Set the default subsystem (optional).
3. Create a symbol file if you are going to use symbolic addresses (see Chapter 5)
4. Start the SIMATIC DDE Server and applications.

4.2.1 Requirement

⚠

Caution

Observe the following with regard to the link between the serial interface of a PC and the programmer interface of the SIMATIC S5 Programmable Logic Controller: The programmer interface of the PLC is always a TTY interface. If the serial interface of the PC (COM1 or COM2) is a V.24 (RS 232) interface, you **must** use a V.24/TTY converter for the link. If you do not, **damage to your PC or PLC hardware** will result.

We recommend:
PG – COM Cable, Art No. 9359 # 1534, AG(TTY)<=→PC(V24)

4.2.2 Installing the Hardware Driver

Basic approach

To install the hardware driver, you have to enter a suitable line with a text editor in your *winstart.bat* file located in the Windows directory. The driver is then automatically activated during MS Windows run up.

1. Open the *winstart.bat* file in your Windows directory. Create a new file if one does not already exist.
2. Insert the line containing the statement for loading the driver into your *winstart.bat* file. The syntax of the entry is described below.

Entry syntax

PATH**SER511 /IOINT=**IRQ **[/IOBASE=**ADR**] [/SWINT=**X**]**

The expressions in square brackets may be omitted. In this case the corresponding default settings become effective.

You can overwrite the default settings by setting the parameters yourself to suitable values. After the example, the significance of the parameters and their default values are explained.

Example entry

C:\SIMDDE\SER511 /IOBASE=2F8 /IOINT=0B

The default setting of SWINT is applied in this particular instance.

Note

Note, when you are setting parameters, that all values are interpreted as hexadecimal numbers.

Setting parameters

/IOBASE=*ADR*

Specify for *ADR* the port address of the serial interface.

If you do not set /IOBASE, the driver entry for serial interface COM1 applies.

/IOINT=*IRQ*
Specify for *IRQ* the interrupt number used with character exchange. This parameter is mandatory.

You arrive at the interrupt number by adding a value of 8 to the interrupt and then calculating the value as a hexadecimal number.

Example:

Interrupt used:	IRQN 5
Interrupt number:	5 + 8 = 13 = D (hexadecimal)
Value to be entered:	D

/SWINT=*X*
Specify for X the hexadecimal number of the software interrupt used to parameterize the driver.
The default value is 80.

Operating several interfaces

The driver is capable of operating up to **four serial interfaces** concurrently. I.e. you can connect a maximum of four automation systems using this driver.

Specify a /IOINT parameter for every interface:

Example

```
C:\SIMDDE\SER511  /IOBASE=2F8  /IOINT=0B   /IOINT=0C   /IOINT=0D
```

The IOBASE parameter must be specified once only, i.e. for the first serial interface. Switching from one interface to another is made possible by subsystem assignments.

Every interface forms a subsystem. Subsystems have to be made known to the DDE Server by either

- an entry on the symbol file, or
- the *susbsys* command.

The subsystem entry in a symbol file is described in Appendix A; you will find the *subsys* command in Chapter 9.

4.2.3 Setting the Interface as the Default Subsystem

Procedure

To define the AS 511 interface as the default subsystem:

1. Start the DDE Server – for example, by activating the appropriate icon on the MS Windows user interface.
2. In the menu bar of the DDE Server main window choose
   Options ▸ Configuration...
   The “Configuration” dialog box is displayed.
3. Click on the drop button in the Drivers field and select the driver **as511.drv** from the opened drop–down list. The name of the driver is then visible in the viewing section.
4. Click on the OK button.
5. If you have changed the default DLL, exit from the DDE Server by choosing **File ▸ Exit** and restart it.

4.2.4 Starting MS Windows, SIMATIC DDE Server, and Application

1. First restart MS Windows for the changes you made to the *winstart.bat* file to become effective.
2. Start the SIMATIC DDE Server.
3. Start your application(s).

4.3 Operation with the SIMATIC S5 PLC via Computer Connection RK 512 (for SIMATIC PC DDE Server Windows 3.11 only)

What you need to do

To start the SIMATIC PC DDE Server Windows 3.11 with a SIMATIC S5 Programmable Logic Controller via a CP 524 or CP 525 board and the computer connection RK 512, you must take the following action:

1. Set the default subsystem (optional).
2. Create a symbol file if you are going to use symbolic addresses (see Chapter 5).
3. Start the SIMATIC DDE Server and your applications.

4.3.1 Requirements

A separate hardware driver is not required for the RK 512 interface.

The interface is controlled by means of the *rk512.drv* driver DLL, which is supplied with the SIMATIC PC DDE Server Windows 3.11.

Several actions have to be programmed on the automation system for the DDE conversation. They include:

- initializing the interface
- periodic calling of the SEND ALL and RECEIVE ALL data handling blocks.

Note: The communication is based upon the 3964R protocol. The default settings of this protocol are used.

Further details will be found in the COM525 Manual, Volume 2/2, Part7, Using Data Handling Blocks, section 2.4.

4.3.2 Setting the Interface as the Default Subsystem

Procedure

To define the RK 512 interface as the default subsystem:

1. Start the DDE Server – for example, by activating the appropriate icon on the MS Windows user interface.
2. In the menu bar of the DDE Server main window choose **Options** ▸ **Configuration...**
   The “Configuration” dialog box is displayed.
3. Click on the drop button in the Drivers Field and select the driver **rk512.drv** from the opened list. The name of the driver is then visible in the viewing section.
4. Click on the OK button.
5. If you have changed the default DLL, exit from the DDE Server by choosing **File** ▸ **Exit** and restart it.

Operating several interfaces

The driver is capable of operating up to **four serial interfaces** concurrently. I.e. you can connect a maximum of four automation systems using this driver.

Every interface forms a subsystem. Subsystems have to be made known to the DDE Server by either

- an entry in the symbol file, or
- the susbsys command.

For this you have to set the interrupt, baud rate, and IO addresses.

The subsystem entry in a symbol file is described in Appendix A; you will find the *subsys* command in Chapter 9.

4.3.3 Starting the SIMATIC DDE Server and Application

1. Start the SIMATIC DDE Server.
2. Start your application(s).

4.4 Operation with the SICOMP S5 SlotPLC (for SIMATIC PC DDE Server Windows 3.11 only)

What you need to do

To start the SIMATIC PC DDE Server Windows 3.11 with a SICOMP S5 SlotPLC, you must take the following action:

1. Install the hardware driver for communication with the SlotPLC.
2. Set the default subsystem (optional).
3. Create a symbol file if you are going to use symbolic addresses (see Chapter 5).
4. Start the SIMATIC DDE Server and your applications.

4.4.1 Requirement

The SICOMP S5 SlotPLC board must be inserted and initialized (refer to the SlotPLC manual /**1**/).

4.4.2 Installing the Hardware Driver

Basic procedure

To install the hardware driver, you have to enter a suitable line with a text editor in your *winstart.bat* file located in the Windows directory. The driver is then automatically activated during MS Windows run up.

1. Open the *winstart.bat* file in your Windows directory. Create a new file if one does not already exist.
2. Insert the line containing the statement for loading the driver into your winstart.bat file. The syntax of the entry is described below.

Entry syntax

PATH**DPR511** **/IOINT**=IRQ **[/IOBASE**=ADR**]** **[/SWINT**=X**]**

The expressions in square brackets may be omitted. In this case the corresponding default settings become effective.

You can overwrite the default settings by setting the parameters yourself to suitable values. After the example, the significance of the parameters and their default values are explained.

Example entry

```
C:\SIMDDE\DPR511     /IOBASE=300   /IOINT=0D
```

The default setting of SWINT is applied in this particular instance.

Note

Note, when you are setting parameters, that all values are interpreted as hexadecimal numbers.

Setting parameters

/IOBASE=*ADR*

Specify for *ADR* the port address of the dual–port RAM.

If you do not set /IOBASE, the default address of 300 applies.

/IOINT=*IRQ*
Specify for *IRQ* the interrupt number used with character exchange in conjunction with the SlotPLC. This parameter is mandatory.

You arrive at the interrupt number by adding a value of 8 to the interrupt and then calculating the value as a hexadecimal number.

Example:

Interrupt used:	IRQN 5
Interrupt number:	5 + 8 = 13 = D (hexadecimal)
Value to be entered:	D

/SWINT=*X*
Specify for X the hexadecimal number of the software interrupt used to parameterize the driver.
The default value is 80.

Operating several SlotPLCs

The driver is capable of operating up to **four SlotPLCs** concurrently. I.e. you can connect a maximum of four automation systems using this driver.

Example

```
C:\SIMDDE\DPR511  /IOBASE=2F8  /IOINT=0B
```

The IOBASE parameter must be specified once only.

The IOINT parameter must similarly be specified once only, since all connected SlotPLCs are operated with the same interrupt number.

Switching from one interface to another is made possible by subsystem assignments.

Every interface forms a subsystem. Subsystems have to be made known the DDE Server by either

- an entry in the symbol file, or
- the *susbsys* command.

A subsystem has to be declared for every SlotPLC.

The subsystem entry in a symbol file is described in Appendix A; you will find the *subsys* command in Chapter 9.

4.4.3 Setting the Interface as the Default Subsystem

Procedure

To define the AS 511 interface of the SlotPLC as the default subsystem:

1. Start the DDE Server – for example, by activating the appropriate icon on the MS Windows user interface by double clicking.
2. In the menu bar of the DDE Server main window choose **Options** ▸ **Configuration...**
   The "Configuration" dialog box is displayed.
3. Click on the drop button in the Drivers field and select the driver **as511.drv** from the opened drop–down list. The name of the driver is then visible in the viewing section.
4. Click on the OK button.
5. If you have changed the default DLL, exit from the DDE Server by choosing **File** ▸ **Exit** and restart it.

4.4.4 Starting MS Windows, SIMATIC DDE Server, and Your Application

1. First restart MS Windows for the changes you made to the *winstart.bat* file to become effective.
2. Start the SIMATIC DDE Server.
3. Start your application(s).

Symbolic Addressing 5

Purpose

The symbol file is an ASCII file that is required for the symbolic addressing of process variables. On the symbol file, you can assign a symbolic name to every logical address in an automation system.

The symbol file is read in by the DDE Server upon startup. You can then use the symbolic names as items in DDE links.

This chapter describes:

- the contents of the symbol file
- the syntax of variables and subsystem declarations

Further information

For information on loading and changing symbol files please refer to Chapter 7 or to the online help.

In this chapter

5.1 General Notes on Creating a Symbol File

Contents of symbol file

The symbol file contains:

- declarations of variables
- declarations of subsystems (optional)
  Declarations of subsystems are required only if you wish to access more than one automation system.

Filename

The filename of a symbol file must have the extension ‘.sig‘.

For example, the simulation driver uses the supplied symbol file called *plcsig.sig*. Three variables are defined in it: ‘triangle‘, ‘array‘ and ‘sawtooth‘.

Required information

To create a symbol file, you require the following information about the variables, whose values are to be accessed using the DDE Server:

- Names of the variables (freely selectable).
- Data types of the variables (see Table 5-1 for valid types).
- Corresponding addresses in the automation system.

Authorized characters

The following rules apply to the symbol file:

- The symbol file must contain only printable ASCII characters.
- Control and special characters, excepting the underscore, are not permitted.
- The first character in a name must be a letter or an underscore, ‘_‘.
- No distinction is made between upper and lower case in variable names.

Keywords

Keywords must always be written with **lower – case letters**. The keywords on the symbol file include the data types

float, **long**, **short**, **byte**, **rbyte**, **lbyte**, **zeit**, **zaehler, merker**, **string**

as well as the expressions **unsigned**, **const** and **subsys**.

Note

The names of system items (see Appendix B) are not permitted as identifiers in variable declarations.

Comments

Comments on the symbol file are preceded – as in the C programming language – by the string ‘**/***‘ and terminated by ‘***/**‘:

/* Comment */

Comments may be several lines long and located anywhere on the symbol file.

5.2 Declarations of Variables in the Symbol File

Syntax

A declaration of a variable in the symbol file has the following simplified syntax:

Example

DataType VariableName=Address;

short triangle=DB[1][1];

short	is the data type
triangle	is the name of the variable
DB[1][1]	is the address (data block 1; data word 1)

Data types of variables

You can see the data types, their applications, and their length from the following table.

Table 5-1 Data types

Data Type	Applications	Length
float	SIMATIC floating-point variable	32 bits
long	Fixed-point variable	32 bits
short	Fixed-point variable	16 bits
byte	Fixed-point variable	8 bits
rbyte*	Fixed-point variable, right byte	8 bits
lbyte*	Fixed-point variable, left byte	8 bits
zeit	SIMATIC time value	16 bits
zaehler	SIMATIC counter value	16 bits
merker	Memory marker value	16 bits
string	String	8 bits per character

* The *rbyte* and *lbyte* data types are the left and respectively right byte of a value. They can be used only in address areas on which data words can be addressed – for example, in data blocks.

float

The data type declares a 32–bit SIMATIC floating–point variable.

Range of values: -10^{38} to -10^{-38}; 0; 10^{-38} to 10^{38}

<u>Examples</u>

“1.2” “3.45678” “3.37E + 05“

long

The data type declares a 32-bit fixed-point variable, with or without sign.

Range of Values		From	To
	signed	– 2 147 483 648	2 147 483 647
	unsigned	0	4 294 967 293

short

The data type declares a 16-bit fixed-point variable, with or without sign.

Range of Values		From	To
	signed	– 32 768	32 767
	unsigned	0	65 535

byte

The data type declares an 8-bit fixed-point variable, with or without sign.

Range of Values		From	To
	signed	– 128	127
	unsigned	0	255

rbyte and lbyte

The data types address the less significant byte (rbyte) and the more significant byte (lbyte) in a word. A word consists of two bytes. Both data types support 8-bit values, with or without sign.

The *rbyte* and *lbyte* data types can be used only in address areas on which data words can be addressed – for example, in data blocks.
Range of values: as for the byte data type

zeit

The data type supports time specifications in string form and in xxx.y format:

xxx is the time value in a range from 0 to 255.
y is the resolution:

Value for y	Resolution
0	0.01 seconds
1	0.1 seconds
2	1 second
3	10 seconds

<u>Examples</u>

27.1	2.7 seconds (27 x 0.1)
37.2	37 seconds (37 x 1)
6.3	60 seconds (6 x 10)

Variables of the zeit type are BCD coded if CF_TEXT format is used. If CF_BYTESTREAM format is used (refer to Appendix C.2.2), they are BCD coded in all address areas except the address area **T** (time) where they are binary coded. The memory areas are listed on page 5-7.

zaehler

The *zaehler* data type declares a 16–bit SIMATIC counter variable.

Range of values: 0 – 999

Variables of the *zaehler* type are BCD coded if CF_TEXT format is used. If CF_BYTESTREAM format is used (refer to Appendix C.2.2), they are BCD coded in all address areas except the address area **Z** (counter) where they are binary coded. The memory areas are listed in Table 5-2, page 5-7.

merker

The data type declares a 16-bit SIMATIC S5 bit memory variable.

string

The data type declares a character string. The length of the string is specified by means of a variable array (see further below). A single character is 8 bits long.

Sign of variables

All fixed-point variables are signed.
If you place the keyword **unsigned** in front of the data type of a variable, the variable is interpreted as being unsigned.

The **zeit**, **zaehler** and **merker** data types are always **unsigned**.

The **float** data type is always **signed**.

<u>Example</u>

unsigned byte m11=DB[10][1];

Read-only variables

By placing the keyword **const** before a variable type, you define a read-only variable. Only read access to such variables is possible.

<u>Example</u>

const unsigned byte wz1=DB[10][1];

Arrays of variables

You declare arrays of variables by specifying the array size in square brackets following the name of the variable.

Arrays are not allowed for variables of the *rbyte* and *lbyte* types or for bit variables.

<u>Example</u>

short array[5]=DB[10][5];

This definition declares an array of length 5 in data block 10, from data word 5 onward. Every single item in a field is 2 bytes long (‘short‘ type).

Addresses of variables

The address comprises the:

- memory area (address area) and
- address value

Memory areas

The authorized memory areas of an automation system (PLC) are shown in the following table:

Table 5-2 Memory Areas

Memory Area	Significance
DB, DX*	Data blocks, extended data blocks
E	Process image input table
A	Process image output table
T	Times
Z	Counter (zaehler)
M	Memory marker
P , Q*	Access to peripherals or extended peripherals
BS*	Operating system data (interpretation dependent upon the PLC connected)
AS*	Absolute addresses

* DX, Q, BS and AS are supported only in SIMATIC S5.

Times and counter values are stored in binary code in memory areas T and Z; they are BCD coded in all other areas.

Address values

Specify address values **in square brackets** directly following the memory areas concerned:

short ein=E[10];

The variable *ein* denotes 2 bytes (short) in the process image input table from the tenth byte onward.

Addressing methods

The following table shows the methods of addressing in the different memory areas.

Memory Area	Method of Addressing
DB, DX*	word
E	byte
A	byte
T	a time value
Z	a counter value
M	byte
P or Q*	byte
BS*, AS*	word

* DX, Q, BS and AS are supported only in SIMATIC S5.

A word consists of a left byte (lbyte) and a right byte (rbyte).

In address areas DB and DX the address value of the required data word is specified in square brackets following the name of the data block:

Example

short byte1=DB[10][4];

This specification addresses a 2-byte long variable (short) in data block 10, data word 4.

Bit addressing

In the areas DB, DX, DE, E, A, and M, you can address a bit within a variable. This option is restricted to variables of types *byte*, *lbyte* and *rbyte*.

Specify immediately after the address value a dot, ‘.‘, and the number of the bit you require.

The least-significant bit is No. 0, the most significant bit is No. 7.

<u>Example</u>

lbyte bit3=DB[10][5].3

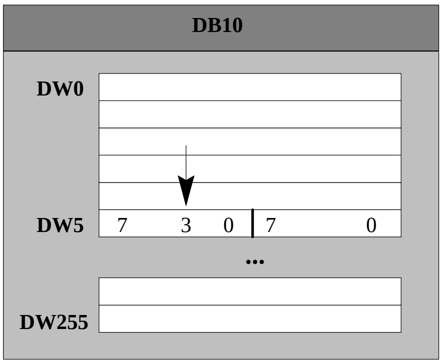
The bit3 variable contains the value of the fourth bit in the left byte of data word 5 in data block 10.

Read:

Variable value = 0   means   bit is not set.
Variable value = 1   means   bit is set.

Write:

Variable value = 0   means   bit will be reset.
Variable value → 0   means   bit will be set.



5.3 Subsystem Declaration on the Symbol File

Not required if...

If only one automation system is connected, there is no need for you to specify a subsystem declaration. In this case the default subsystem defined in the driver is used.

Required if...

If several automation systems are connected, you have to specify an additional address, i.e. the subsystem name. The subsystem name specifies the automation system from which the value of a variable originates. An automation system is permanently assigned to a subsystem.

Example

The driver for the serial interface, **as511.drv**, can serve several interfaces. Precisely one automation system, which then forms a subsystem, is connected to every serial interface.

Assigning to a subsystem

Every variable on the symbol file is assigned precisely to a subsystem. The following assignments are possible:

- default subsystem
- one subsystem for several variables
- one subsystem for one variable

Default subsystem

If a subsystem has not been specified on the symbol file, the ‘**default**‘ subsystem defined in the driver is used.

For example, the ‘default‘ subsystem is assigned to the COM1 interface in the case of the ‘as511.drv‘ driver.

Subsystem for several variables

A subsystem declaration applies to all the following definitions of variables on the symbol file until another subsystem is declared.

Subsystem for one variable

You can also specify the subsystem as part of the address on the symbol file. To do this, place the subsystem name in front of the usual address and separate the address with a dot, ‘.‘:

unsigned short m11u_short=PLC1.DB[10][1];

The variable gets its value from subsystem PLC1, data block 10, data word 1.

Syntax

The syntax of the subsystem declaration is:

subsys Name={par1,par2,par3,par4, ‘dllname‘};

subsys	keyword for declaration of a subsystem
Name	name of the subsystem. The name is freely selectable – for example, PLC1, PLC2.
par1 – par4	maximum of four parameters; their significance depends on the subsystem concerned.
dllname	name of the driver DLL

Necessary parameters
The names of the different driver DLLs and the corresponding parameters are described in Appendix A.

Absolute Addressing 6

Purpose

Absolute addressing can be used by Windows applications in the item part of DDE calls in place of symbolic variable names. With absolute addressing, you can thus access through the DDE Server values in the automation system for which variables have not been defined on the symbol file.

If you work only with variables that have been defined on the symbol file, you can skip this chapter.

Syntax

An instruction with absolute addressing is structured as follows:

s5addr(CPUNo;OpType;OpNo;Start;Length;Format)[SubsysName]

The instruction name **s5addr** applies for SIMATIC S5 and S7/M7.

Separator

The different parts of the address string are separated by a semicolon ‘**;**‘.

CPUNo

Range of Values		Significance
1 – 4	for SIMATIC S5	Number of the CPU
1	for SIMATIC S7/M7	

Address Areas (OpType)

The **OpType** represents the address area. The address areas that are specifically supported depend on the automation system addressed.

Range of Values	Significance
db	Data blocks
dx*	Extended data blocks
mb	Memory marker
eb	Process image input table
ab	Process image output table
pb	Peripheral area
qb*	Extended peripheral area
zb	Counter
tb	Times
bs*	Operating system data area
as*	Absolute addresses

* dx, qb, bs and as are supported only in SIMATIC S5.

OpNo, Start and Length

You can see the authorized range of values of the address specifications, i.e. op. no., start and length from the following table.

OpType	OpNo	Start	Length
db	1...255 for S5/S7 0...65 535 for M7	0...4095 for S5/S7 0...32 766 for M7	1...4096 for S5/S7 1...32 767 for M7
dx*)	1...255	0...4095	1...4096
mb	0	0...255 for S5/S7 0...65 534 for M7	1...256 for S5/S7 1...65 535 for M7
eb	0	0...511	1...512
ab	0	0...511	1...512
pb	0	0...65 533	1...65 534
qb*)	0	0...255	1...256
zb	0	0...511	1...512
tb	0	0...511	1...512
bs*)	0	0...511	1...512
as*)	0	0... 65 535	1 ... 65 536

*) only for SIMATIC S5.

Format

Format	Significance	Example
KM	Binary representation	00000000 00000000
KT	Time representation	000.0 to 999.9
KY	Every byte decimal	101
KZ	Counter in BCD	000 to 999
KF	Short	−32768 to 32767
KG	32-bit float (IEEE)	+1.234E+3
KC	String	abcde

Note

The SIMATIC DDE Server **does not check at runtime** whether the specified address areas do in fact exist on the automation system.

Subsystem name

The subsystem name must be defined on the symbol file or be set by means of a command (cf. Chapters 5 and 9).

If you do not specify a name with absolute addressing, the ‘default‘ subsystem is used.

Example

An absolute address in the form

ddeplc|default|s5addr(1;DB;10;5;2;KF)

would read on CPU 1, in data block 10, data word 5, two words and return them in short format.

Operating the DDE Server

7

Introduction

This chapter describes how to:

- start and exit the DDE Server
- load and modify a symbol file
- retrieve status information
- manage connections and variables.

For further information on this topics see the online help. The configuration of the DDE Server and the trace function are described in Chapter 8.

In this chapter

7.1 Starting the DDE Server

Starting options

There are different ways of starting the DDE Server:

- Activating the appropriate DDE icon.
- Entering a command on the command line of Program Manager.
- Activating a symbol file or the executable file of the DDE Server in the File Manager or in the Explorer respectively.
- Activating the “Start” button under Windows 95.

DDE icon

Double-click the left mouse button on the DDE Server icon to activate it. The DDE Server is started and uses the default symbol file.

Command line

Choose in Program manager the “Run File“ function. A dialog box, in which you can enter the pathname of the DDE Server and, optionally, a symbol file as parameters, is then opened.

File Manager or Explorer

Double-click on the file containing the symbol file (filename: *.sig) to open it. The DDE Server is then started and uses the symbol file you selected.

Alternatively:

Open the executable file **ddeplc.exe** of the SIMATIC DDE Server in the File Manager or the Explorer.

Starting under Windows 95

Click on the “Start” Button and select the DDE Server in its program group.

Start once only!

The DDE Server can run only once under Windows. If you attempt to start the DDE Server while it is running, a corresponding message is output to the screen.

Automatic start

If you continually need the SIMATIC DDE Server, copy the icon into the “Autostart” group in MS Windows.

Set the Program Manager (File→Properties) so that the SIMATIC DDE Server is started as an icon.

7.2 Exiting the DDE Server

Precaution

If you want to exit from the DDE Server while DDE conversations to other applications are still open, a corresponding message box is displayed.
You then have two options of proceeding:

- Cancel exiting from the Server (by clicking on the ‘Cancel‘ button), or
- Close the connections and exit from the Server (by clicking on the ‘OK‘ button).

Dialog

The procedure for when the main window of the SIMATIC DDE Server is open on the screen is:

- Choose the menu command **File→Exit**

If connections to other applications are still open, then the dialog box mentioned above is displayed. Otherwise you exit from the Server without any other message or acknowledgement.

Icon

The procedure for when the Server is running in the background, i.e. the SIMATIC DDE Server icon is on the desktop, is:

1. Select the icon.
2. Choose ‘Close‘ on the drop-down menu.

Here again, the DDE Server displays a message box about any connections that are still open.

Exiting from Windows

When you exit from Windows and the Server is still active, a message is displayed if any connections are still open (as above). If not, you exit from the Server without any message being displayed.

7.3 Overview of the Menu Functions of the DDE Server

Menu Functions

This section provides you with an overview of the menu functions of the DDE Server. The uses of the individual functions will be described in the chapters that follow. For more detailed information see the online help.

Once the Server has been started the main window is opened.

The menu bar contains four menus – ‘File‘, ‘Options‘, ‘Status‘, and ‘Help‘.

File

At this point you will find the following functions:

Open...	Load a symbol file
Edit...	Change the current symbol file with a text editor
Exit	Exit from the DDE Server

The functions associated with the symbol file are described in Section 7.5.

Options

At this point you will find functions for configuring the Server and controlling trace behavior.

Detailed information about this point will be found in Chapter 8.

Status

When you choose this menu, you are provided with status information about the Server, the current connections, and the variables, and you can manage connections and variables.

Managing connections and variables is described in Sections 7.6 and 7.8.

Help

At this stage you can call up general information on the DDE Server and the Help system for the DDE Server.

Contents	Contents of the online Help for the DDE Server
About...	Release, year of release, and licensee of the DDE Server are displayed.

7.4 Using Online Help for the DDE Server

Windows Help system

The DDE Server Help system makes use of the Windows online Help system. It features glossary-style and context-sensitive help.

The Help system runs in a separate window, thereby accessing the application-specific help file, DDEPLC.HLP, in which the SIMATIC DDE Server Help system is defined.

Call options

You can use online Help in the following ways:

- Activating Help in the menu bar of the main window
- Using function keys
- Pressing the Help button in dialog boxes.

Activating Help→Contents

Procedure

1. On the menu choose **Help → Contents**.
2. Select a keyword on the Contents topic.
   Keywords are underlined. Once a keyword has been selected the text of the corresponding explanation is displayed.

Note

You can have the system directly display instructions on how to use the Help system of MS Windows after it has been called:

1. Choose the question mark, **?**, in the menu bar of the help window.
2. Choose **How to Use Help** on the drop-down menu.

Function Key F1

If you press the **F1 key** and the mouse pointer is located in the main window, the Contents topic of the Help system for the SIMATIC DDE Server is displayed. At this stage you can call up further information on selecting a keyword.

If you press the **F1 key in a dialog bo**x **or when a menu is open**, the Help text for the current dialog box or selected menu item is displayed (context-sensitive help).

Function key SHIFT F1

If you press the **SHIFT F1 key** in the main window of the SIMATIC DDE Server, the shape of the cursor changes to a question mark. The next click shows information on the item you selected.

Example:
If you make a selection – as you normally would do after pressing SHIFT F1 – in the menu bar of the DDE Server, information is displayed on the menu item you selected.

You can exit from SHIFT F1 mode by pressing the **ESC key**.

The SHIFT F1 mode is not active in dialog boxes.

Help button

All dialog boxes of the DDE Server contain a Help button. If you press this button, a Help window containing advice on the input options for that dialog box is opened.

7.5 Loading and Modifying the Symbol File

Introduction

The symbol file is an ASCII file in which the variables and subsystems for the Server are defined. A symbol file is loaded each time the SIMATIC DDE Server is started. While the DDE Server is running, you can, if necessary:

- load a new symbol file or
- modify the current one with an ASCII editor.

The functions for processing the symbol file are located in the ‘File‘ menu the main window of the DDE Server.

Loading the Symbol File

Condition:

There must not be any connections open when a symbol file is loaded. If there are, a corresponding message box is displayed, and you have to close the open links before you can proceed (refer to Section 7.6).

Note

When you load a new symbol file, all existing variables in the DDE Server are deleted.

Procedure

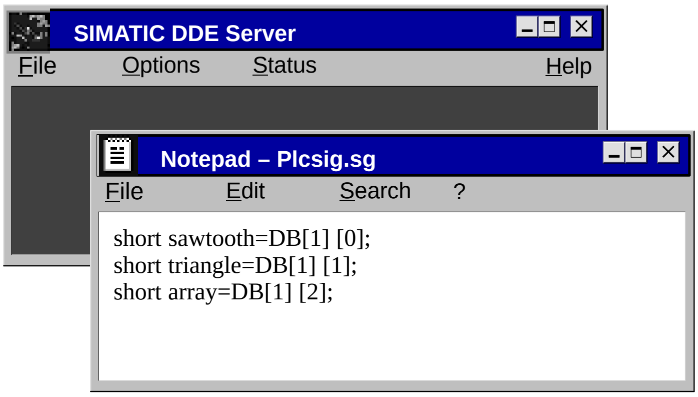
1. Choose the menu command **File** → **Open**. The customary Windows dialog box appears for selecting a file.
2. Click on the file you require and then choose the OK button. If the file containing the symbol file is not displayed, you must first change to the corresponding directory.

Modifying the Current Symbol File

Symbol files are ASCII files that can be modified with any text editor. You can modify the current symbol file of the DDE Server by using a menu.

Procedure

1. Choose the menu command **File** → **Edit**. A text editor is then called for modifying the symbol file.



```
short sawtooth=DB[1] [0];
short triangle=DB[1] [1];
short array=DB[1] [2];
```

2. Enter the changes on the symbol file with the editor.
3. Save the symbol file and exit from the editor. If the symbol file has been modified, you are asked whether you want the new symbol file to be loaded.
4. Click on the ‘OK‘ button to confirm and to load the modified symbol file. Only then do the changes become effective.

Note

Changing the name of the symbol file does not take effect in the DDE Server. Please use the menu command **File** → **Open**, in order to load another symbol file.

7.6 Managing Connections

Introduction

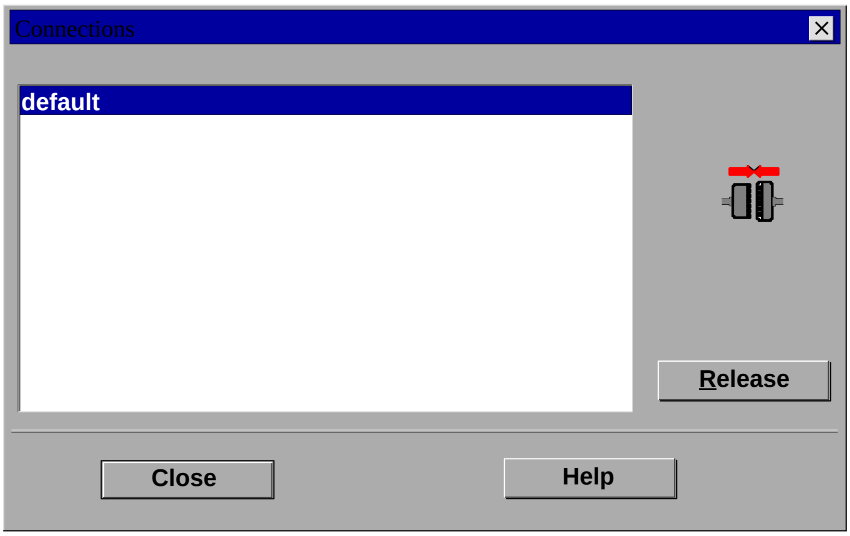
In this chapter you learn how to view the current connections and to release selected connections.

Viewing connections

Procedure:

- Choose the menu command **Status** → **Connections**

  The ‘Connections‘ dialog box is displayed. The field displays all the currently open connections to the DDE Server.



default

Releasing single connections

You can release active connections either singly or collectively.

Procedure:

1. Select the lines you require from the list of connections or select all the connections. The lines are then highlighted.
2. Click on the ‘**Release**‘ button.
3. Click on ‘**Close**‘ button.

Note

You can release no connection as long as hot or warm links are still active on it. You must terminate these links in the client application first.

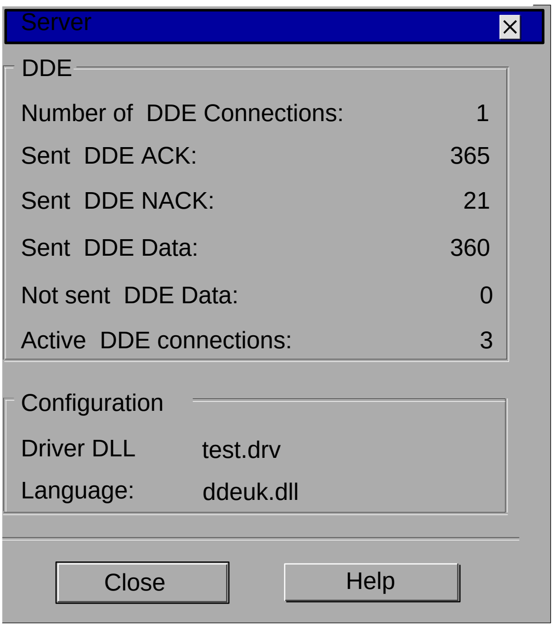
7.7 Retrieving the Status of the DDE Server

Procedure

You can retrieve statistical information on dynamic data exchange (DDE) as well as data on the Server configuration.

- Choose the menu command **Status→Server**.

The status box of the DDE Server is displayed.



Server

DDE

Number of DDE Connections:	1
Sent DDE ACK:	365
Sent DDE NACK:	21
Sent DDE Data:	360
Not sent DDE Data:	0
Active DDE connections:	3

Configuration

Driver DLL	test.drv
Language:	ddeuk.dll

Fields in the status box

By using the status box, you can follow the most important counters of the SIMATIC DDE Server in online mode. The status box contains two panels – ‘DDE‘ and ‘Configuration‘. The fields in these two panels have the following significance:

Field	Description
DDE:	
Number of DDE Connections:	Number of open DDE connections.
Sent DDE ACK:	Number of DDE ACK (positive acknowledgment) messages sent by the SIMATIC DDE Server.
Sent DDE NACK:	Number of DDE NACK (negative acknowledgment) messages sent by the SIMATIC DDE Server.
Sent DDE Data:	Number of sent DDE DATA messages.
Not-sent DDE Data:	Number of discarded DDE DATA messages.
Active DDE Connections:	Number of active DDE hot or warm links (DDE ADVISE).
Configuration:	
Driver DLL	Current driver DLL for the default subsystem (test.drv is the test driver for simulation mode).
Language	Current DLL for the language-specific sections of the SIMATIC DDE Server (ddeger.dll for German; ddeuk.dll for English).

7.8 Managing Variables

Introduction

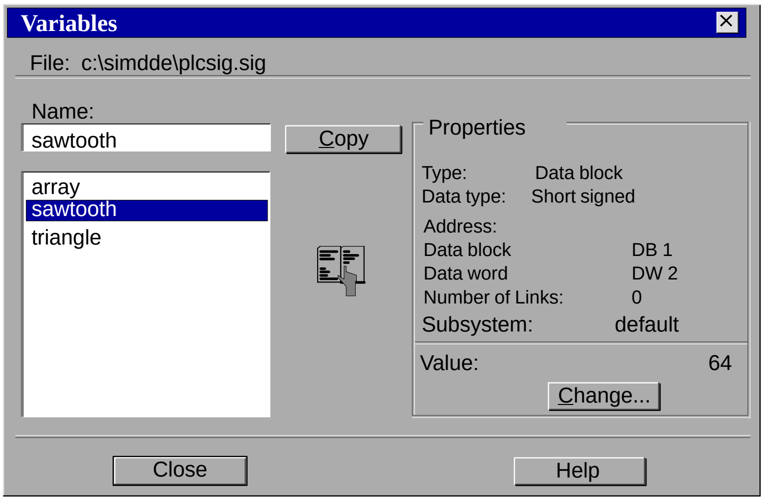
All the variables declared in the symbol file can be comfortably managed on the user interface. You can do the following:

- Read the variables
- Change the value of a variable
- Copy the value of a variable to the clipboard

Reading Variables

Procedure:

1. Choose the menu command **Status** → **Variables....** The 'Variables' dialog box of the DDE Server is displayed.
2. Choose a variable from the displayed list. The name of the variable you selected is highlighted.



File: c:\simdde\plcsig.sig

Name: sawtooth

array
sawtooth
triangle

Properties

Type: Data block
Data type: Short signed
Address:
Data block DB 1
Data word DW 2
Number of Links: 0
Subsystem: default

Value: 64

3. Once a variable has been selected, its name is displayed in the 'Name' field. Information about this variable is displayed in the 'Properties' field.

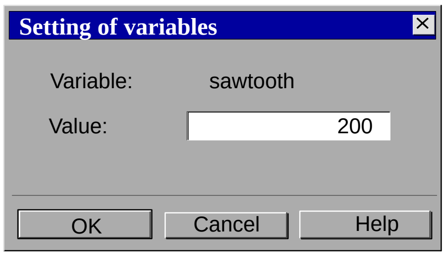
The fields display the following information:

- The 'File' field displays the pathname of the file containing the symbol file.
- The 'Name' field displays the name of the variable you selected.
- The 'Properties' field displays information about the variable.
- The 'Value' field shows the value of the variable at the time you selected it. If you click again on the name of the variable on the list, its value is updated.

Changing Variables

Procedure:

1. Open the ‘Variables‘ dialog box as described in the previous section.
2. Select a variable.
3. Click on the ‘Change...‘ button in the ‘Value‘ field.
   The ‘Setting Values of Variables‘ dialog box is displayed:



Variable: sawtooth

Value: 200

4. Correct the value by means of the keyboard.
5. Click on the ‘OK‘ button. You can immediately see the change in the ‘Variables‘ dialog box.

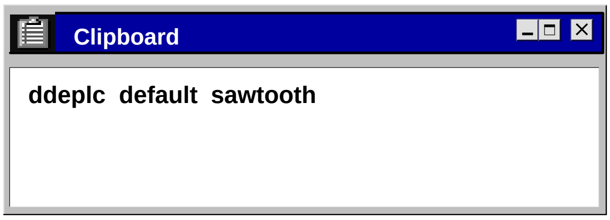
Result: The new value is written to the address area of the automation system.

Copying a Variable

The ‘Copy‘ function lets you open DDE links to standard applications with a minimum of effort.

Procedure:

1. Open the ‘Variables‘ dialog box as described in the previous section.
2. Select a variable.
3. Click on the ‘Copy‘ button. The free end of the link is now put on the Clipboard. If you open the Clipboard, you will see the DDE address of the variable, consisting of the server name (ddeplc), the topic (default) and the name of the variable.



ddeplc default sawtooth

4. Select the position in your application at which you would like to insert the variable – for example, a worksheet cell in Excel.
5. Choose ‘Paste Link‘ in your application.
   The link has now been opened (hot link), and the value of the variable in the application is updated automatically.

8 Configuring the DDE Server

Setting options

You have the possibility of configuring the behavior of the SIMATIC DDE Server in a dialog box. You can set:

- the cycle time for periodic polling of the automation system
- simple modifications to the communication protocol
- the driver for the default subsystem

With the Trace mechanism, you can log the values of variables and function calls for test and troubleshooting purposes.

Note

You should not make changes to the Server configuration until you have become thoroughly familiar with dynamic data exchange (DDE) under MS Windows.

Changes to the parameters in the Configuration dialog box are saved and act as the new default configuration parameters the next time you start the DDE Server.

In this chapter

8.1 Setting the Polling Cycle Time

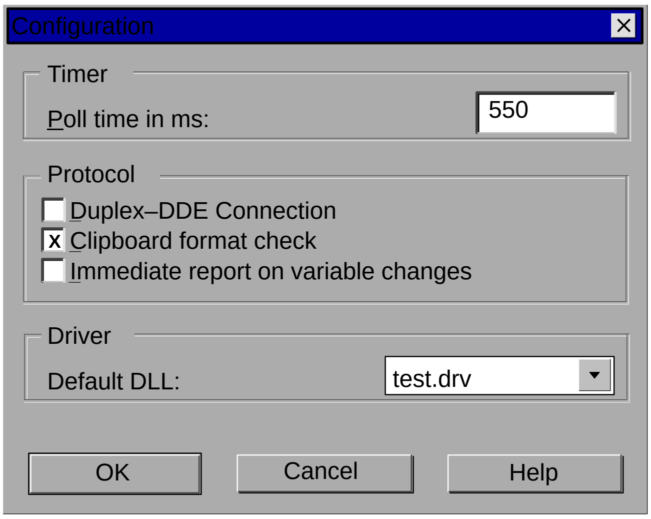
Effect

The entry in the ‘Polling‘field determines the time interval after which the DDE Server reads in those variables from the automation system whose values have to be updated and tests them for changes. If the variable in the connection has changed since it was last read in, its current value is passed to the client (warm or hot link).

Setting

Procedure

1. Choose the menu command **Options** → **Configuration**. The “Configuration” dialog box is displayed.



Configuration

Timer

Poll time in ms: 550

Protocol

☐ Duplex–DDE Connection
☒ Clipboard format check
☐ Immediate report on variable changes

Driver

Default DLL: test.drv

2. Enter a value for the polling cycle time in milliseconds in the corresponding field.

   Range: 55 ms to 65535 ms
   Default setting: 550 ms

Note

If the polling time you select is too short, Windows system may come to a standstill on account of excessive system load. The polling time must be higher or equal to the minimum period of operator-interface cyclic-read jobs supported by the automation system.

If the polling time you select is too long, changes to variables may be lost.

8.2 Setting the Protocol

Setting options

Procedure

- Choose the menu command **Options** → **Configuration**. The “Configuration” dialog box is displayed.

The dialog box presents you with three functions:

- Duplex DDE Connection
- Check Clipboard Format
- Immediate Reporting of Changes to Variables

You activate and deactivate the functions by making the corresponding entry or checking the box (toggle function). When a function has been activated, a check mark appears in the corresponding check box. The following sections provide you with information on when you should (have to) activate and deactivate these functions.

Why Duplex DDE Connections

This function affects handling of DDE ADVISE messages in a DDE conversation.

Many DDE applications transfer internal variable values to DDE servers only when they have been enabled to do just this by a DDE ADVISE request. For such DDE applications, the SIMATIC DDE Server provides a **duplex DDE connection** (duplex hot link), depending on the configuration.

A conversation conforming with the DDE protocol does not provide duplex DDE connections. Duplex DDE connections represent a deviation from the DDE protocol (see below).

- Enable the duplex DDE connections only when your application requires it (first try to do without duplex DDE connections). As a rule, do not select a duplex DDE connection (default setting).

  Range: yes/no
  Default setting: no

Deviation from protocol with duplex DDE connections

The parameters transferred in a DDE ADVISE message are:

- identification of the DDE conversation
- name of the item for linking
- additional information about the values to be sent – for example, the Clipboard format and acknowledgment upon receipt of data

DDE ADVISE messages

A DDE ADVISE message is used by the client to transfer the name of an item to the Server and to request it to send data to the client whenever the value of this item changes. The data may be the value of the item itself or merely a signal to the effect that the value has changed. In this way a permanent connection to the item is opened.

When a duplex DDE connection has been set, the following deviation from the DDE protocol applies:

- The DDE Server answers a DDE ADVISE request from a DDE client with a **second DDE ADVISE request to the client**.
- If the DDE client accepts the DDE ADVISE request from the Server, this means that a duplex hot link has been opened.

From this point of time onwards the DDE client and the SIMATIC DDE Server keep each other automatically informed of any change to the variables specified in the DDE ADVISE request.

Note

This deviation from the protocol is not supported by all DDE-compatible applications. For example, MS Excel ignores a DDE ADVISE request by the Server to the client.

Check Clipboard Format

By deactivating the check, you can improve the performance of a DDE conversation.

With a DDE request, a client always transfers the Clipboard format of the messages sent as a parameter. The SIMATIC DDE Server supports only the CF_TEXT and the CF_BYTESTREAM format. MS Excel, however, sends requests with different Clipboard parameters – including CF_TEXT. This behavior has an adverse effect on DDE conversation performance since the format is checked every time a DDE request is received and may be rejected.

If you de-activate **Check Clipboard Format**, the SIMATIC DDE Server immediately returns the data – without checking the format – in CF_TEXT format. CF_TEXT format is supported by most MS Windows applications.

- Activate Check Clipboard Format if you wish to prevent invalid data formats from being transferred. If the application does not detect CF_TEXT format in returned data, reception of these DDE messages will be rejected.

  If your application does detect CF_TEXT format, you can leave the format check deactivated for performance reasons.

  Range: yes/no
  Default: yes (Check Clipboard Format on)

Note

A conversation using MS Excel macros works only if the Clipboard format check has been activated.

Immediate Reporting of Changes to Variables

For performance reasons, the SIMATIC DDE Server does not report changes to variables to a DDE client when the change to the variable (DDE POKE) has occurred on the DDE channel of an existing link (DDE ADVISE) to the specified variable.

To ensure that the DDE Server abides by the protocol, the “**Immediate Reporting of Changes to Variables**” option was added to the **Configuration** dialog box.

If this option has been set on the configuration dialog box, changes to the value of a variable for which there is a link (DDE ADVISE) are explicitly reported to the writing DDE client (DDE DATA) after the variable has been changed on the same DDE channel (DDE POKE).

If this option has not been set, changes to variables are reported only to DDE clients linked to the variable, but not to the writing client. In this case, the acknowledgment (DDE ACK) of the write instruction is confirmation for the writing client of the variable having been changed.

- Change this default setting only if your application requires it.

  Range: yes/no
  Default: no (no immediate reporting)

8.3 Setting the Default Subsystem and Completing the Configuration

Setting the Driver DLL

Procedure

1. Choose the menu command **Options** → **Configuration**. The ¿Configuration¿ dialog box is displayed. You can select the driver for the default subsystem in the ‘Driver‘ field.
2. Click on the drop-down button adjacent to the output field and select a driver from the drop-down list.

   Default: test.drv (simulation driver)

Completing the configuration

To complete your configuration and for it to become effective:

1. Choose the OK button.

   If the driver you set is different from the previous one, a dialog box is opened, prompting you to restart the DDE Server, if you want the changes to take effect.
2. Choose the OK button.
3. Exit the DDE Server by choosing **File**→**Exit** and restart it.

8.4 Using the Trace Function

Overview of Trace functions

With the Trace mechanism, you can log the values of variables and function calls for test and troubleshooting purposes. You can call the following functions by choosing **Options** in the menu bar of the main window:

- Activate Trace
- Reset Trace
- Set Trace options

What you need to do

Usually you will do the following:

1. Deactivate Trace, if it happens to be active with **Options**→**Trace active**.
2. Open the dialog box for configuring Trace with **Options**→**Trace**.
3. Select **Trace Items – DDE**, and also **Date** and Time under **Trace Format**.
4. Select **Trace Output – File**.
5. Specify the path and filename of the Trace file if you do not wish to apply the default setting.
6. Activate Trace with **Options**→**Trace active** and then wait a while.
7. Deactivate Trace by deactivating the menu **Options**→**Trace active**.
8. Inspect the Trace file.
9. Choose **Options**→**Reset Trace**.

Trace file

Example of entries in a Trace file

```
07.04.94      09:40:35         XTYP_REGIST  <ddeplc:(2a54)→
07.04.94      09:40:35         XTYP_REGISTER        BaseServiceName
ddeplc        InstServiceName  ddeplc:(2a54)
07.04.94      09:40:35         XCLASS_NOTIFICATION
ddeplc:(2a54)>:
```

Explanations of the names and registers can be found in **/4/**.

8.5 Trace Configuration

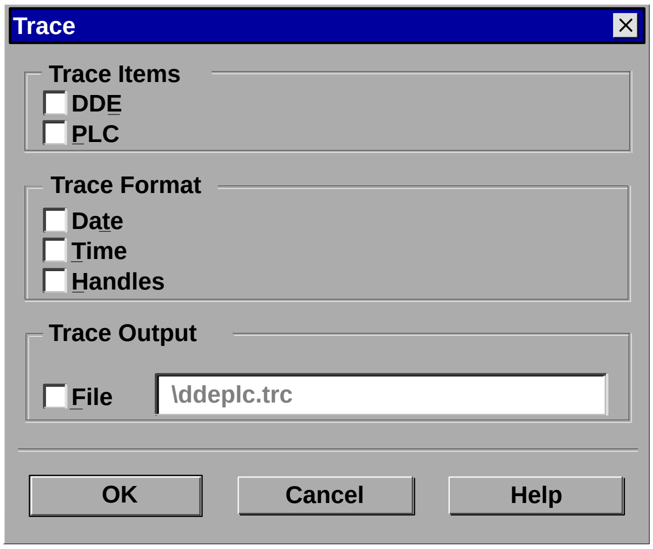
Requirement

Before activating Trace, you should first set the Trace configuration options.

Procedure

You can set the log format and the name of the Trace file using the user interface.

1. Choose **Options→Trace**.
   The ‘Trace‘ dialog box is displayed.



2. Enable or disable the options you require (toggle function). Enabled options are selected by checking the check boxes. The functions are described below.
3. Then close the dialog box by clicking on the ‘OK‘ button. The changes you made to the Trace configuration will then become effective.

DDE Trace Items

Enable ‘Trace Items – DDE‘ if you require all DDE messages to be logged.

PLC Trace Items

Enable ‘Trace Items – PLC‘ if you require all the requests of the automation system and their acknowledgments to be logged.

Date Trace Format

Enable ‘Trace Format – Date‘ if you require the Trace output to include the date.

Time Trace Format

Enable ‘Trace Format – Time‘ if you require the Trace output to include the time of day.

Handles Trace Format

Enable ‘Trace Format – Handles‘ if you require the Trace output to include all the Windows handles of the DDE client and the DDE Server.

Trace Output

Enable ‘Trace Output – File‘ if you require the complete Trace text to be written to the specified file. The Trace file is an ASCII file that can be inspected with a text editor. You can change the pathname of this file, too.

The pathname is dimmed if the Trace output is disabled.

Note

You cannot switch to the Trace output **File** if the Trace is active. You can open a new file only after deactivating and then re-activating the Trace (see below).

Writing to a file

To write Trace information to a file, you have to:

1. First choose the Trace output **File** on the **Trace...** configuration mask.
2. Then activate Trace mode by choosing **Options**→ **Trace active**.

Trace information is then written to the file if you deactivate Trace mode again.

⚠ **Caution**

The trace information is appended to the Trace file as long as the Trace is active. The contents of the Trace file is deleted only when the Trace is reset.

Do not allow the Trace file to grow too large.
(Refer to ‘Resetting Trace‘)

8.6 Activating, Deactivating and Resetting Trace

Activating and De-activating Trace

Procedure

- Choose **Options** → **Trace active**

‘Trace active‘ is a toggle function:
If a check mark is not visible in front of the lettering on the option menu, Trace is deactivated.

If you choose the function, Trace is activated and a check mark is visible. The Trace system is now active and Trace information is logged.

Trace is deactivated by choosing it again, and the check mark disappears.

If you require Trace information to be written to a file, then first select **Trace Output – File** in the **Trace...** dialog box, and then activate Trace mode.

Resetting Trace

Procedure

- Choose **Options** → **Trace reset**

Trace is reset, and the complete contents of the Trace file are deleted.

Commands and Options

9

Introduction

From this chapter you learn:

- How commands are executed by the DDE Server
- Which options can be set by the DDE Server.

In this chapter

9.1 Commands

Commands are also referred to as ”pseudo–variables”. Commands are always passed at the third addressing level, i.e. **item** under the **default** topic:

ddeplc default *Command*

The following commands can be executed by the DDE Server:

- poll – set or retrieve the polling cycle time
- absolut – set reporting of values of the variable in every polling cycle or retrieve the current setting
- load – load a symbol file
- subsys – declare a subsystem

poll

With this command you can set or retrieve the polling cycle time for the current connection.

For setting the polling cycle time, you have to pass the time value in ms as an argument. The syntax is as follows:

poll=*time*

The *time* parameter may assume values ranging from 55 ms to 65 535 ms. (55 ms is the lowest timer unit in MS Windows). The value is rounded off to the next integer multiple of the polltime value set by the menu **Options** → **Configuration**

If you specify 0 as the value, polling is disabled for that connection. Changes to variables in existing hot and warm links can no longer be reported to the client.

For retrieving the current polling cycle time, the **poll** command without argument must be executed with the DdeRequest call.

absolut

Sets reporting of values of the variable in every polling cycle or retrieves the current setting.

For setting the reporting of variables, the syntax is as follows:

absolut=*value*

The command causes the SIMATIC DDE Server to report the value of a variable to the client in every polling cycle for warm and hot links, even if it has not changed. In the case of writing with 0 as a value, this characteristic is reset. A value not equal to 0 enables the characteristic.

For retrieving the current setting, the **absolut** command without argument must be executed with the DdeRequest call.

load (*path*)

With this command you can load a symbol file. To do this, you have to specify the complete pathname of the file containing the symbol file.

Example
Load a symbol file using Excel macros:

=POKE (ChanNum;”&load(c:\simdde\signew.sig)”;A4)

(in this case the third argument ‘A4‘ is not evaluated)

or

=EXECUTE(char;”load(c:\simdde\signew.sig)”)

subsys *arguments*

With this command you can declare a subsystem. The parameters are the same as for the subsystem declaration on the symbol file.

Example
Declare the subsystem with Excel macros:

=POKE (ChanNum;”&subsys (plc1, par1, par2,par3,par4,as511.drv)”;A4)

or

=EXECUTE(ChanNum;”subsys (plc1, par1, par2,par3,par4,as511.drv)”)

Note

The subsystem declared in this manner is not significant in respect of the variables on the symbol file. You can specify it only in absolute addresses.In the examples above the third argument ‘A4‘ of the POKE macro is not evaluated.

Command Names

To distinguish them from variables of the same name, commands are always prefixed by an ampersand ‘(**&**)‘.

Examples and Return Values

You can execute DDE commands by using the function calls DdeExecute DdeAdvise and DdePoke. The following table shows examples of the function calls and their return values.

Command	Examples	Return Values
poll		

Set the poll cycle time	rw_bool = DdeExecute(...,”poll=110”)
	rw_bool= DdeAdvise(...,”&poll=330”,...)
	rw_bool= DdePoke(...,”&poll=220”,...)
		rw_bool= True/False

Retrieve the actual poll cycle time	rw_int =DDERequest(...,”&poll”, rw_buffer,...)	rw_int = Number of bytes rw_buffer = Actual poll cycle time
absolut		

Enable/disable reporting	rw_bool = DdeExecute(...,”absolut=1”)
	rw_bool = DdeAdvise(...,”&absolut=0”,...)
	rw_bool = DdePoke(...,”&absolut=0”,...)
		rw_bool= True/False

Retrieve the actual setting	rw_int = DDERequest(...,”&absolut”, rw_buffer,...)	rw_int = Number of bytes
		rw_buffer = Actual setting

load		

Load symbol file	rw_bool = DdeExecute(..., ”load(c:\plcn.sig)”)
	rw_bool = DdeAdvise(..., ”&load(c:\plcn.sig)”,...)
	rw_bool = DdePoke(...,”&load(c:\plcn.sig)”,...)
		rw_bool= True/False

	rw_int = DDERequest(...,”&load(c:\plcn.sig)”,rw_buffer”,...)	rw_int = Number of bytes rw_buffer = Name of the actual symbol file
subsys		

Declare a subsystem	rw_bool = DdeExecute(...,”subsys(as1,0,2,0,0,mpi.drv)”)
	rw_bool = DdeAdvise(...,”&subsys(as1,0,2,0,0,mpi.drv)”,...)
	rw_bool = Depone(...,”&subsys(as1,0,2,0,0,mpi.drv)”,...)
		rw_bool= True/False

	rw_int = DDERequest(..., ”&subsys(as1,0,2,0,0,mpi.drv)”, rw_buffer,...)	rw_int = 0
		rw_buffer = empty

You can retrieve the result of the command execution by using the system item RETURNMESSAGE.

9.2 Options

Options define certain characteristics of the DDE connection. They are passed at the second addressing level (default).

poll=*time*

Sets the polling cycle time.

This means that you can set the poll rate (cycle time for the current connection in ms (55 ms is the lowest timer unit in MS Windows). The *time* parameter may assume values ranging from 55 ms to 65535 ms. The value is rounded off to the next integer multiple of the polltime value set by the menu **Options** → **Configuration**

If you specify 0 as the value, polling is disabled for that connection. Changes to variables in existing hot and warm links can no longer be reported to the client.

absolut

Sets reporting of values of the variable in every polling cycle.

The option causes the SIMATIC DDE Server to report the value of a variable to the client in every polling cycle for warm and hot links, even if it has not changed.

Examples

ddeplc | default(poll=55) | triangle

Reads a variable with a polling cycle of 55 ms.

ddeplc | default(absolut) | triangle

Reads a variable in every polling cycle. The default value applies to the polling cycle time.

Example Conversation with MS–Excel 10

Introduction

This chapter gives you examples of DDE communication with MS Excel using macros and formula.

In this chapter

10.1 Types of Conversation with MS Excel

Types of conversation

In MS Excel, a distinction is made between

- explicit and
- implicit conversations.

Explicit conversation

In an **explicit** conversation, a high – level language or a macro is used to open a channel, to send requests, and to close the channel again.

Implicit conversation

In an **implicit** conversation, a DDE reference (service, topic, or item) is named from within an application, the program then opening a channel without the user being able to see it, retrieving the value of the item, and likewise closing the channel again. Far easier to use, an implicit conversation establishes a hot link to the application it talks to.

Macro language

All common Windows applications support DDE. The description of the DDE protocol, however, defines only the **parameters** of the DDE commands, not the **macro language** for calling the DDE commands. This is the reason why the commands differ from one application to another.

The user interface of the SIMATIC DDE Server consequently defines only the significance of the individual DDE parameters for the SIMATIC DDE Server.

By way of the macro language of MS Excel, examples are shown in the following of DDE calls in a Windows application.

10.2 MS Excel DDE Command Macros

Basic sequence of events

A DDE conversation between Windows applications takes place in a logic channel. The Excel macro function
chan=INITIATE(ParameterList)
opens a channel to a application that supports DDE. The channel number is the return parameter of the macro required during the DDE conversation in order to reference the logic channel.

For exchanging data and other services, MS Excel features the macro functions **POKE**, **REQUEST** and **EXECUTE**. The first two macros can be used to send data to the other DDE application or to receive data from the other DDE application. You use the EXECUTE macro to send commands to the other DDE application. The format of the data is invisible to the user. It is negotiated by MS Excel and the SIMATIC DDE Server in the background.

The DDE channel is closed again with the
TERMINATE(chan) macro function.

Macro list

The following DDE command macros are available in MS Excel:

INITIATE

TERMINATE

REQUEST

POKE

EXECUTE

Note

These macros can be used with Excel V.4 and V.5.

Detailed descriptions of MS Excel macros will be found in the “!Microsoft Excel” manual.

10.3 Examples with MS Excel DDE Command Macros

Opening a channel

ChanNum=INITIATE (”ddeplc”;”default”)

Opens a connection to the default topic on which data can be retrieved and sent and commands executed.

Closing a channel

=TERMINATE(ChanNum)

Closes the connection with the ID ”ChanNum”.

Retrieving data

=REQUEST(ChanNum;”triangle”)

Supplies the value of the ”triangle” variable.

Sending data

=POKE(ChanNum;”triangle”;A7)

Assigns the value of cell A7 to the ”triangle” variable.

Executing commands

=EXECUTE (ChanNum;”load(c:\simdde\signew.sig)”)

Loads a new symbol file.

Example file

A macro model with DDE macros will be found in the root directory of the SIMATIC DDE Server under:

\ddeplc02.xlm

Note

A conversation using MS Excel macros works only if the Clipboard format check has been activated.

10.4 DDE Hot Link with MS Excel Formulas

Implicit conversation

Hot links represent a rapid and effective method of integrating the values of variables in other applications into the current environment with constantly having to retrieve them every time.

Hot link with macros

The establishment of hot links with macros is **not possible in MS Excel**.

Hot link with formulas

A hot link, i.e. an uninterrupted link, can be established in MS Excel only with the help of remote references (reference to an application in another program).

Remote links are particularly easy to handle in MS Excel and are implemented as formulas. A formula calculates the contents of a cell. Unlike macros, which extend over several cells, formulas are clearly arranged and are specified within a cell.

Read access

Read access to the ”triangle” example variable is executed from a MS Excel form by entering the following formula (”hot link”), for which a cell array is selected.

=ddeplc|default!triangle

The above formula is activated by simultaneously pressing the three keys “Control”, “Shift“, and “Enter“.

The above formula causes MS Excel to send the SIMATIC DDE Server (”ddeplc”) an INITIATE request of its own accord. When the link has been opened, MS Excel subsequently sends a request to the DDE Server for a hot link to be created to the **triangle** variable. When the Server has accepted the request, MS Excel initializes the cell containing the hot link formula by means of a REQUEST request to the DDE Server. In the event of an error, MS Excel displays an error.

Write access

Write access to a variable is **not possible using MS Excel formulas.**

Example file

A MS Excel worksheet with DDE formulas will be found in the root directory of the SIMATIC DDE Server under:

\ddeplc01.xls

Subsystem Declarations A

Introduction

The driver DLL is parameterized by the *subsys* declaration on the symbol file. The syntax of this declaration is as follows:

subsys *name*={*par1, par2, par3, par4, ‘dllname‘*};

name is the symbolic name of the interface, user-defined.

par1 to par4 are the transfer parameters Parameter 1 to 4, which are described in the sections that follow.

dllname is the name of the driver DLL.

This appendix describes the transfer parameters for the following driver DLLs:

- MPI.DRV
- AS511.DRV
- RK512.DRV
- TEST.DRV

In this chapter

A.1 Transfer Parameters for MPI.DRV

Parameter

The transfer parameters define the address of the module to which a connection is to be set up via the Multipoint interface.

Parameter	Description	Default
Parameter 1:	Device ID: Determines the type of communication: 0 → Multipoint interface (MPI) 1 → Communication bus (C bus) 2 → Local Only the device ID is allowed with the SIMATIC PC DDE server Win 3.11	0
Parameter 2:	Host ID*	2
Parameter 3:	Rack Slot ID*	0
Parameter 4:	Remote Host ID*	0

* The semantics of this parameter depends on the type of station (SIMATIC-300 or SIMATIC-400) you communicate with. and on the value parameter 1 (device ID).

Host ID

Parameter 2 (host ID) defines either the MPI node address of the CPU or function module (FM), or the communication bus address (rack slot number) in the S7/M7-400 system (e.g. when the DDE server is running on an FM 456-4, see example 2).

In the case of local addressing, parameter 2 defines where local communication will occur; the MPI node address of the module in the S7/M7 300 system and the communication bus address (rack slot number) in the S7/M7 400 system.

Rack Slot ID

Parameter 3 (rack slot ID) defines the communication bus address (rack slot number) in the S7/M7-400 system. In the S7/M7 300 system parameter 3 always has the value 0.

Remote Host ID

Parameter 4 (remote host ID) is used only with the SIMATIC M7 DDE Server Windows95 when addressing a module in a remote system S7/M7-300/400. It defines the following:

- MPI node address of the CPU or function module (FM) in the remote system S7/M7-300
- MPI node address of the CPU in the remote system S7/M7-400

Note

If the SIMATIC M7 DDE Server Win 3.11 runs on an FM 456-4 function module, only the CPU and other function modules within the same SIMATIC-400 station can be accessed. This constraint does not apply to the SIMATIC M7 DDE Server Win 95.

Examples

The following examples show subsystem declarations for SIMATIC S7/M7 automation systems. The driver DLL is mpi.drv

1. In the following example, a subsystem is declared (**mpisub10**), which sets up a connection to the communication partner using the MPI node address 10 (host ID). The driver DLL is **mpi.drv**.

   subsys mpisub10={0,10,0,0,‘mpi.drv‘};

2. In the following example, a subsystem is declared (**mpisub8**), which sets up a connection to the communication partner via the communication bus (within a SIMATIC S7/M7 station) using the communication bus address 8 (rack slot). The driver DLL is **mpi.drv.**

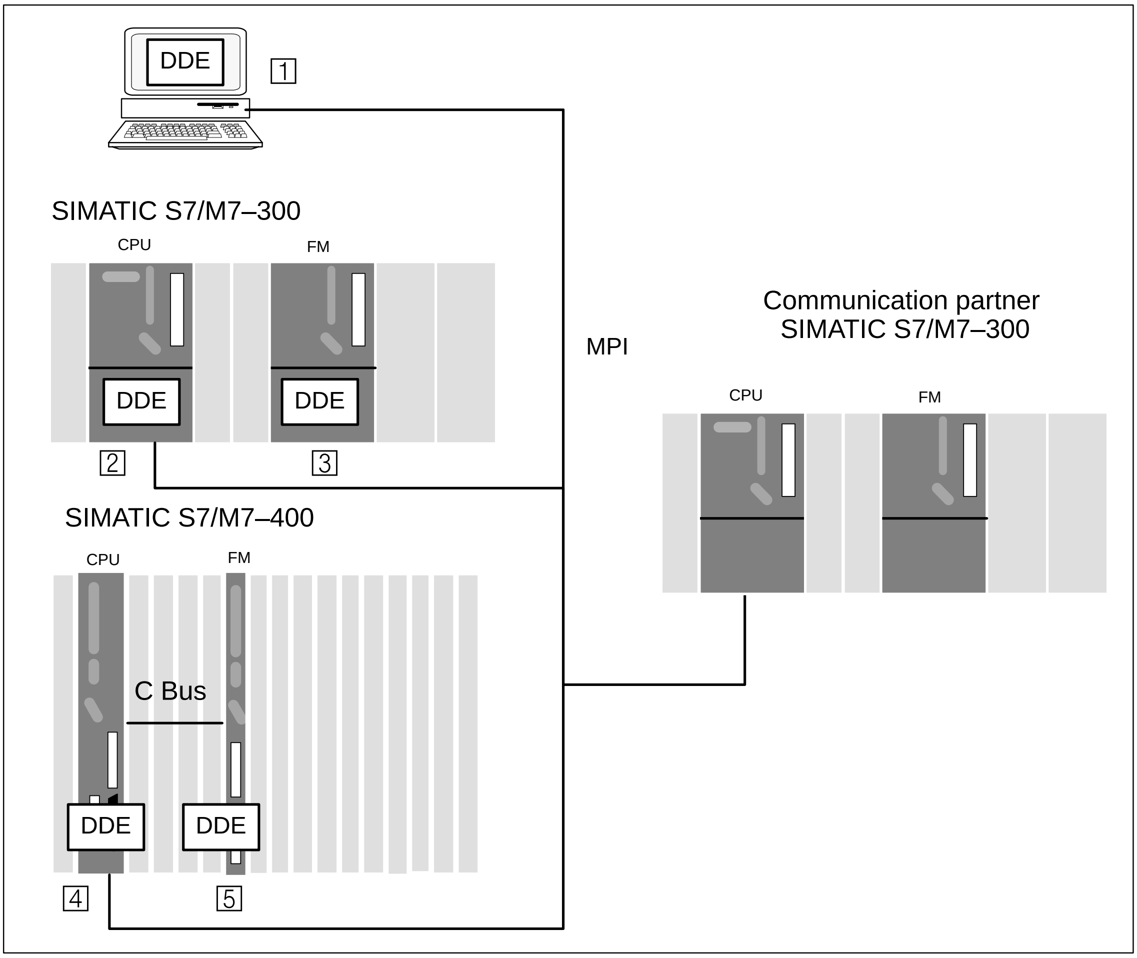
   subsys mpisub10={1,8,0,0,‘mpi.drv‘};

3. In the following example, a subsystem is declared (**mpisub3_7**), which sets up a connection to the node via the Multipoint interface with the communication bus address 7 (rack slot) in the communication bus subnet of a CPU with the MPI node address 3. The driver DLL is **mpi.drv**.

   subsys mpisub10={0,3,7,0,‘mpi.drv‘};

Communication with SIMATIC S7/M7-300

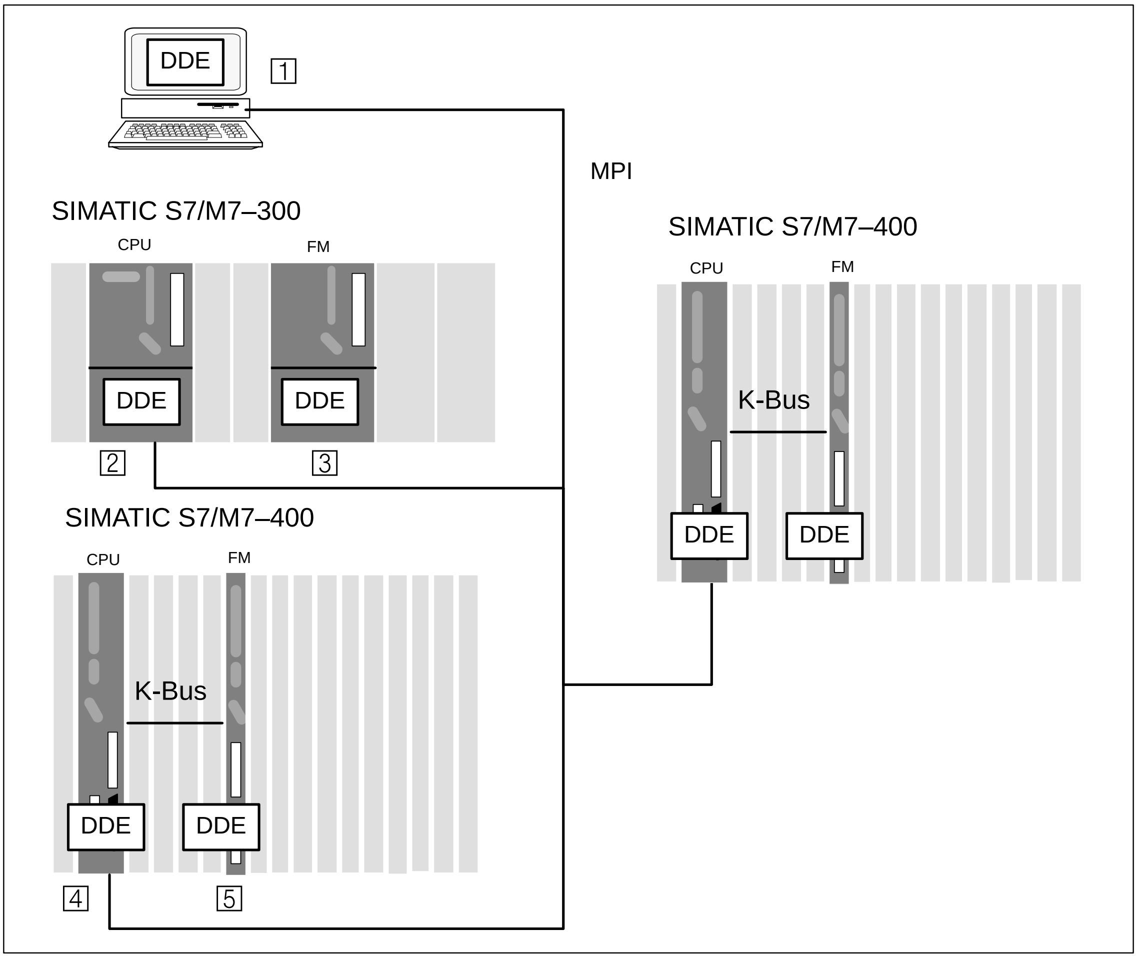
Parameter significance for the subsystem declaration. Communication partner is a CPU or FM in a SIMATIC-300 station.



Parameter	DDE Server runs on		
	PC, or M7-300 CPU/FM 1 2 3	**M7-400 CPU** 4	**M7-400 FM** 5
Par 1 Device ID	0	0	1
Par 2 Host ID	MPI node address of destination CPU/FM	MPI node address of destination CPU/FM	Rack slot ID of CPU in the own 400 station
Par 3 Rack Slot ID	0	0	0
Par 4 Remote Host ID	0	0	0 or MPI node address of CPU/FM in the remote 300 station (DDE-Server Win95 only)

Communication with SIMATIC S7/M7-400

Parameter significance for the subsystem declaration. Communication partner is a CPU or an FM in a SIMATIC-400 station.



Parameter	DDE Server runs on		
	PC, or M7-300 CPU/FM [1] [2] [3]	M7-400 CPU [4] [5]	M7-400 FM [4] [5]
Par 1 Device ID	0	0	1
Par 2 Host ID	MPI node address of destination CPU/FM	MPI node address of remote CPU or Rack Slot ID of FM in the own 400 station	Rack slot ID of CPU/FM in the own 400 station
Par 3 Rack Slot ID	0 or Rack Slot ID of destination 400-FM	0 or Rack Slot ID of FM in the remote 400 station	0 or Rack Slot ID of FM in the remote 400 station (DDE-Server Win95 only)
Par 4 Remote Host ID	0	0	0 or MPI node address of CPU/FM in the remote 300 station (DDE-Server Win95 only)

Local Communication

Parameter semantics for the subsystem declaration for local communication on SIMATIC M7-300/400 CPU/FM with MPI.DRV

Parameter	DDE Server runs on	
	300 CPU/FM	400 CPU/FM
Par 1 Device ID	2	2
Par 2 Host ID	own MPI node address	own Rack Slot ID
Par 3 Rack Slot ID	0	0
Par 4 Remote Host ID	0	0

A.2 Transfer Parameters for AS511.DRV

Parameters

The transfer parameters are the same as for the DOS TSR:

Parameter	Description
Parameter 1:	I/O address of the serial interface
Parameter 2:	Software interrupt:
	The value has to be the same as the value specified as /SWINT in the *winstart.bat* file (refer to Chapter ???); If /SWINT is not specified in *winstart.bat*, Parameter 2 has to be set to 0.
Parameter 3:	Is ignored
Parameter 4:	Is ignored

Default

If the DLL is installed as the default DLL, the default settings from the installation of DOS TSR are applied.

Example

In the following example, a subsystem called **df15**, installed at I/O address 0x2e8, is declared. The driver DLL is **as511.drv**.

```
subsys df15={0x2e8,0,0,0,‘as511.drv‘};
```

A.3 Transfer Parameters for RK512.DRV

Parameters

The transfer parameters define the behavior of the DLL for a CP 524/CP 525.

Parameter	Description	Default
Parameter 1:	I/O address of the serial interface	0x3f8
Parameter 2:	Interrupt request You have to specify values between 0 and 7 (interrupt requests 0 to 7) for interrupts on the master interrupt controller and values between 8 and 15 (interrupt requests 8 to 15) for interrupts on the slave controller	4
Parameter 3:	Baud rate of the link 1200 bd =0 2400 bd= 1 4800 bd= 2 9600 bd= 3 19200 bd= 4 38400 bd= 5 115200 bd= 6	3 (9600 Bd)
Parameter 4:	Priority (refer to COM 525 manual) Low= 0, High= 1	0 (low)

Note

The communication is based upon the 3964R protocol. Default protocol settings are used.

For a DDE conversation, the application has to call the **SEND ALL** and **RECEIVE ALL** functions (refer to COM 525 manual, Volume 2/2, Part 7, Using Data Handling Blocks, Section 2.4).

Example

In the following example, a **cp524** subsystem, installed at I/O address 0x2e8 and interrupt 7 on the master interrupt controller, is declared. The baud rate is set to 19200 bd, the priority is low. The DLL driver is **rk512.drv**.

subsys cp524={0x2e8,7,4,0,‘rk512.drv‘};

Constraint

The CPU number can be set at the present time only by absolute addressing. This is planned as an extension to the symbol file.

A.4 Transfer Parameters for TEST.DRV

Parameters

There are no transfer parameters.

Default

Not applicable

Example

In the following example, a subsystem called **test** is declared for the **test.drv** driver DLL.

```
subsys test={0,0,0,0,‘test.drv‘};
```

System Items and Non-System Items Supported by the DDE Interface

B

Introduction

This appendix describes the standard system items and non-system items supported by the DDE Server

In this chapter

B.1 System Items

Under the **system** topic, the SIMATIC DDE Server supports the items mentioned in this section for retrieving general information via the DDE interface.

These items represent the third component of the DDE address after the **ddeplc** service and the **system** topic:

ddeplc system *system item*

System Item	Description
SYSITEMS	generates a list of all items that can be retrieved with the system topic.
TOPICS	generates a list of all supported topics.
STATUS	specifies the status of the SIMATIC DDE Server (BUSY/ READY).
FORMATS	generates a list of supported Clipboard formats.
HELP	specifies current information about the SIMATIC DDE Server.
TOPICITEMLIST	generates a list of all items supported by the SIMATIC DDE Server for the **system** topic.
SELECTION	generates a list of all the variables for which a hot link exists.
RETURNMESSAGE	returns additional information about a transaction request. The significance of return values is explained in the next section, .

Note

The names of system items are not permitted as identifiers in variable declarations in the symbol file.

B.2 Return Values of the RETURNMESSAGE System Item

Identification	No.	Significance
E_DDE_SUCCESS	0	The request was successfully completed.
E_DDE_MEMORY	1	The DDE Server cannot allocate any more memory.
E_DDE_PLC_BUSY	2	Internal (synchronization) error of the DDE Server.
E_DDE_NOT_IMPL	3	The DDE Server cannot execute this request in its current version.
E_DDE_VAR_NOT_EXIST	4	The DDE Server does not recognize the variable you specified.
E_DDE_UNKNOWN_TYPE	5	The DDE Server cannot interpret the type of variable you specified.
E_DDE_INVALID_STRING_FORMAT	6	Invalid transfer format of a variable. Refer to the details of the different data types.
E_DDE_WRONG_SYSTEM_ITEM	7	The system item you specified does not exist.
E_DDE_VAR_IS_NOT_ADVISED	9	A DDE UNADVISE was made to a variable not linked to a DDE ADVISE.
E_DDE_TOO_MUCH_PARAMS	10	The function call in DDE Execute or DDE Request contains too many parameters.
E_DDE_COMMAND_SYNTAX	11	Syntax error in the function call or in the command in DDE Execute or DDE Request – for example, too few parameters.
E_DDE_COMMAND_WRONG_BLANK	12	The command contains an invalid blank.
E_DDE_COMMAND_WRONG_DELIMITER	13	A command contains a wrong separator.
E_DDE_UNKNOWN_COMMAND	14	The DDE Server does not know the command.
E_DDE_CONNECTION_NOT_EXIST	15	The DDE Server cannot find the DDE link you specified.
E_DDE_ALREADY_ADVISED	16	The variable you specified has already been linked.
E_DDE_TOO_MUCH_PARAMETER	17	There are too many parameters in the function call.
E_DDE_CONNECTION_ALREADY_EXIST	18	DDE link management error.
E_DDE_CREATE_CONN_STRUCT	19	The DDE Server cannot open any further DDE links.
E_DDE_EMPTY_LIST	20	DDE Server internal error.
E_DDE_ELEM_NOT_IN_LIST	21	DDE Server internal error.
E_DDE_CREATE_DDEVAR_STRUCT	22	The DDE Server cannot create a DDE variable structure.
E_DDE_INVALID_EXECUTE_STRING	23	The execute string of the DDE Execute command is invalid.
E_DDE_CF_FORMAT	24	The only Clipboard format supported by the DDE Server is CF_TEXT.

Identification	No.	Significance
E_DDE_INV_POKE_DATA	25	The poke string of the DDE Poke command is invalid.
E_DDE_WRONG_DDE_APP_NAME	26	Service not known.
E_DDE_WRONG_TOPIC	27	Topic not known.
E_DDE_PLC_ABORT	28	Communication with PLC aborted.
E_DDE_WRONG_FORMAT	29	Invalid format (only CF_TEXT allowed).
E_DDE_READ_ONLY	30	Variable must not be written.
E_DDE_NO_SUBSYS	33	Creation of subsystem no longer possible.
E_DDE_WRONG_DLL_STRUCT	34	The **dllinit** initialization function of the driver DLL was not found.
E_DDE_WRONG_DLL	35	Loading of the driver DLL not possible (the Server searches only in the root directory of the DDE Server).
E_DDE_SUBSYS_EXIST	36	Subsystem already exists.
E_DDE_WRONG_SUBSYS	38	Wrong subsystem name in absolute addressing.
E_DDE_WRONG_VARLIST	39	Syntax error on the symbol file.
E_DDE_CONV_EXIST	40	A new symbol file cannot be loaded because there are still some links open.
E_DDE_WRONG_SIGLI	41	Wrong name or path of the symbol file.
E_DDE_VAR_EXIST	42	Variable already exists in the symbol file.
E_DDE_READ_ERR	43	Variable read error
E_DDE_WRITE_ERR	44	Variable write error
E_DDE_CONNECT_ERR	45	Connect error
E_DDE_INVALID_CYCLE_TIME	46	The poll cycle time set for a hot link is not supported by the automation system
E_DDE_MAX_PDU_SIZE	47	The transferred amount of data exceeds the PDU size settled with the automation system

B.3 Non – System Items

Under the **default** topic, the SIMATIC DDE Server supports the items mentioned in this section for retrieving general information about the DDE interface.

These items represent the third component of the DDE address after the **ddeplc** service and the **default** topic:

ddeplc default *non-system item*

TOPICITEMLIST generates a list of all the items supported by the SIMATIC DDE Server under the **default** topic.

FORMATS generates a list of supported Clipboard formats for this topic.

Note

The names of non-system items are not permitted as identifiers in variable declarations in the symbol file.

Technical Data and Transfer Formats C

Introduction

This appendix describes:

- the technical data of the DDE Server and
- the transfer formats supported by the DDE Server

In this chapter

C.1 Technical Data

Hard disk requirements

Product	Space required
SIMATIC PC DDE Server Win3.11:	1.2 MB
SIMATIC M7 DDE Server Win3.11:	1.2 MB
SIMATIC M7 DDE Server Win95:	0.5 MB

Main memory configuration

at least 4 MB

RAM requirements

Purpose	Space required
for code	about 250 KB for runtime
for connections	500 bytes for management structures (once only)
for variables	200 bytes per variable

No. of variables and connections

The number of variables and connections is limited only by the memory available in MS Windows.

malloc implementation

Storage space is requested by malloc. Malloc implementation under Windows creates global buffers. The number of global buffers allocated with malloc is limited to 4000 and is independent of the main memory configuration.

Amount of data with MPI

When using DDE conversations via MPI, the amount of transmitted data depends on the PDU (Process Data Unit) size settled with the communication partner. If the amount of data sent or requested exceeds this size, the call is rejected. On SIMATIC M7, you can retrieve the PDU size using the M7 API function call **M7GetPDUSize**.

C.2 Data Transfer Formats

When a connection is opened by a DDE client, a format for transferring data has to be specified in every case.

The SIMATIC DDE Server can process data in CF_TEXT and CF_BYTESTREAM formats.

C.2.1 CF_TEXT

CF_TEXT corresponds to the standard MS Windows format. The data types possible on the symbol file are interpreted as follows:

Data type	Interpretation
short	decimal, possibly signed
long	decimal, possibly signed
byte	decimal, possibly signed
lbyte, rbyte	decimal, possibly signed
float	IEEE compatible (cf. C function printf())
zeit	0.0 to 999.9
zaehler	000 to 999
merker	0000 0000 0000 0000
string	string of declared length

Field separator

Individual items in fields are separated from each other by the string ‘\r\n‘ (in line with the conventions of the CF_TEXT format).

C.2.2 CF_BYTESTREAM

Registration

Before first using a data transfer format, it must be registered with Windows by means of the Windows API function call **RegisterClipboardFormat**. This applies to CF_BYTESTREAM as well.

uint_value= RegisterClipboardFormat(format_name)

The *uint_value* return value is the registration no. of the data transfer format. It is passed as a format ID argument to further DDE transactions

Variable length

Data type	Description	Length
float	IEEE floating – point variable	32 bit
long	Fixed – Point variable	32 bit
short	Fixed – Point variable	16 bit
byte	Fixed – Point variable	8 bit
lbyte	Fixed – Point variable, left byte	8 bit
rbyte	Fixed – Point variable, right byte	8 bit
zeit	SIMATIC time value	16 bit
zaehler	SIMATIC counter value	16 bit
merker	SIMATIC bit memory	16 bit
string	string of characters	8 bit per character

Field separator

No field separators are transmitted. When data is read, an additional byte containing the access status (success=0, failure!=0) is appended to the beginning. The length of the transmitted data and therefore the length of the allocated buffer depends on the data type.

Examples:

When reading two long values, 9 bytes are transmitted, i.e. 1 byte access status and 8 bytes of data.

When writing two long values, 8 byte of data are transmitted.

Note

You must use the Windows API function calls **DdeAccessData** and **DdeUnaccessData** in order to access the complete transferred data, since the function call **DdeGetData** reads data only until the first occurrence of 0x0.

Read and write access

Please mind the following when reading or writing data in CF_BYTESTREAM format:

Data type	Read	Write
short, zeit, zaehler, merker	When a 16 – bit value is read, it is converted from the SIMATIC byte order (high byte, low byte) to the Intel byte order (low byte, high byte).	When a 16 – bit value is written, it is converted from the Intel byte order (low byte, high byte) to the SIMATIC byte order (high byte, low byte).
zeit (time)	Binary access, if this data type is defined in the T (time) memory area BCD access, if this data type is defined in another memory area, e.g. DB.	
zaehler (counter)	Binary access, if this data type is defined in the Z (counter) memory area BCD access, if this data type is defined in another memory area, e.g. DB.	
zeit and zaehler	Data is supplied in the SIMATIC byte order upon reading. The bytes are suitably adapted to Intel byte order (refer also to ‘Berger: Automatisieren mit SIMATIC S5 – 115 U, (Automating with the SIMATIC S5 – 115 U, sections 2.3 and 2.4‘).	Upon writing time data, bits 0 to 9 (time value) and bits 12 to 13 (time reference) are meaningful Upon writing counter data, bits 0 to 9 (counter value) are meaningful.
long	When a 32-bit value (2 words, 4 bytes) is read, it is converted from the SIMATIC byte order (high byte, low byte) to the Intel byte order (low byte, high byte).	When a 32-bit value (2 words, 4 bytes) is written, it is converted from the Intel byte order (low byte, high byte) to the SIMATIC byte order (high byte, low byte).
float	As with long data type above. A conversion from KG format to IEEE-float format is executed. As with long data type above.	A conversion from IEEE-float format to KG format is executed.
Bit variables		Bit variables are interpreted as bytes. Values not equal to 0 correspond to “1“.

DDE Block Transfer

D

O/I system requirements

High-performance **Operator Interface** (O/I) systems make high throughput demands on the process data interface. The DDE protocol has been enhanced specifically for this case to include the option of block transfer.

The block transfer mechanism of the DDE Server makes it possible to connect an operator interface system in a standardized manner to automation systems.

In this chapter

D.1 Communication between Server and Application (O/I System)

Definition of block transfer

With a block transfer, **any number** of variables can be read from the automation system with just **one** DDE request to the Server. The variables may be located in random data areas on the PLC.

The performance benefits of block transfer are based on the saving of the system overhead, which arises upon transfer of specific variables.

Server function

The DDE Server makes sure that the variables are read and sent back to the application in a single block.

Single – variable access

The Server reads specific variables in succession with a separate request to the automation system.

Advantages of DDE Block Transfer

The combination of DDE block transfer and single – variable access provide the O/I system with the following advantages:

- Lists of variables can be created dynamically.
- The only variables that have to be read by the PLC are the ones that are currently required.
- The application programmer does not have to make an additional data area available on the PLC for storing O/I data.
- No additional code is required in the application for supplying data areas for O/I.

Any PLC application can thus supply data for a O/I system. All that is required is knowledge of the distribution of the variables for visualization in the data areas of the automation system.

D.2 Notes Common to All Commands

Overview of commands

The block DDE interface covers the following commands:

- **DefineList**
  Defines a list of variables in the DDE Server.
- **DeleteList**
  Deletes a list of variables in the DDE Server.
- **GetList**
  Reads out the definition of a list of variables from the DDE Server.
- **VarList**
  Command for specific access via a list of variables (DDE Request, DDE Advise, and DDE Poke)
- **Result**
  Retrieve status of the last write access

The commands are described in greater detail from Section D.3 onward.

Command execution

The commands can be executed using:

- DdeExecute (...)
  rw_bool=DdeExecute(...)
- DDERequest (...)
  rw_int=DdeRequest(..., rw_buffer)
- DDE Advise (...)
  rw_bool= DdeAdvise (...)
- DDE Poke (...)
  rw_bool= DdePoke (...)

Since the Excel EXECUTE macro can only return Boolean values, the application should use only commands that can similarly supply Boolean values – for example, DeleteList.

Note

The characteristic of hot links of keeping the value concerned up to date, is used only for the VarList command.

& as command prefix

To distinguish them from the names of variables, command names are prefixed by “&” in the case of DDE Request and DDE Advise. The parameters remain the same in all cases.

Return values

With DDE Request and DDE Advise, the result of command execution is returned. The following table describes the significance of the values returned in the variables *rw_bool*, *rw_int* and *rw_buffer.*

Table D-1 Return Values

Command	**Return Value of**				
	DdeExecute rw_bool	DDERequest		DdeAdvise rw_bool	DdePoke rw_bool
		rw_int	rw_buffer		
DefineList	True/False	0[1]	empty	True/False	True/False
GetList	-	No. of bytes	Names of items in the list of variables	-	-
DeleteList	True/False	0[1]	empty	True/False	True/False
VarList	True/False	No. of bytes	Values of variables in CF_TEXT or CF_BYTESTREAM	True/False	True/False
Result	-	No. of bytes	Write status in CF_TEXT or CF_BYTESTREAM	-	-

[1] You can check the result of a request by retrieving the system item **ReturnMessage** with DdeRequest.

Return formats

Variables can be supplied in two formats:

- CF_TEXT

  In CF_TEXT format, individual values are converted into ASCII and separated by special characters, namely \r\n. The complete string is null – terminated (“\0”).

- CF_BYTESTREAM

  The CF_BYTESTREAM format is special to the DDE Server. To simplify a conversation, the only conversion made is from SIMATIC to Intel byte order (see Section C.2.2).

  Only the commands **VarList** and **Result** can be used with the CF_BYTESTREAM format.

Access to lists of variables

The following table shows the specific behavior of the DDE Server upon reading and writing lists of variables.

<table>
<tr><th>Access Type</th><th>CF_TEXT</th><th>CF_BYTESTREAM</th></tr>
<tr><td rowspan="2">Read</td><td>In the CF_TEXT format individual values are converted to ASCII format and separated from each other by the string ‘\r\n‘. The whole string is null–terminated.
In the case of problems in accessing a variable, its value is set as a string, namely ‘????‘.</td><td>When read, every variable value is preceded by a byte signifying the access status. The values of the access status signify:
=0 Access successful
!=0 PLC–specific error number with access problems
The length of individual variable values is automatically defined by using the data type of the variables on the list of variables.
Bit variables are supplied with a length of one byte.
Strings are filled to their declared length.</td></tr>
<tr><td colspan="2">Warm links/hot links: If either a warm or hot link has been opened, a message is issued if at least one variable on the list has changed.</td></tr>
<tr><td rowspan="2">Write</td><td colspan="2">Following a write access using the list of variables, you can read out the status of the last write access for the list of variables using the **Result** command (refer to Section D.7).</td></tr>
<tr><td>In the case of problems in accessing a variable, its value is set as a string, namely ‘????‘.</td><td>For each variable of the list a byte signifying the access status is returned (refer to the “Read”section of this table for the significance).</td></tr>
</table>

D.3 DefineList Command

Syntax

DefineList(*listname*,*var_list*)

Description

With this command you define a list of variables called *listname*, which consists of the variables listed in *var_list*.

The name of the list of variables can then be used in all DDE commands (hot,cold or warm link; poke) as the item name. A list of variables may contain up to 64 variables.

The name of a variable may be defined on the symbol file of the DDE Server or specified as an absolute address.

Arguments

Argument	Description
listname	Freely selectable name of the list of variables
var_list	List of variables var_list is a string containing variables separated by a comma.

Return values

DDEML Call	Variable	Value
DdeExecute	rw_bool	True/False
DDERequest	rw_int	0[2]
	rw_buffer	empty
DdeAdvise	rw_bool	True/False
DdePoke	rw_bool	True/False

[2] You can check the result of a request by retrieving the system item **ReturnMessage** with DdeRequest.

Errors

If a list of variables contains at least one error, the whole list is rejected. Possible errors are:

- References to symbolic names not defined on the symbol file.
- Invalid absolute addressing.
- Variables are located in different subsystems.
- More than 64 variables were specified

Example

Definition of a list of variables named **m7list** using the function call DdeRequest. The list contains 4 variables named **a**, **b**, **c** and **d**.
DdeRequest(...”&DefineList(m7list,a,b,c,d)“,...)

D.4 DeleteList Command

Syntax

DeleteList(*listname*)

Description

With this command you delete the list of variables called *listname* on the Server. Lists of variables are named and defined with the DefineList command.

Argument

Argument	Description
listname	Name of the list of variables to be deleted

Return values

DDEML Call	Variable	Value
DdeExecute	rw_bool	True/False
DDERequest	rw_int	0[2]
	rw_buffer	empty
DdeAdvise	rw_bool	True/False
DdePoke	rw_bool	True/False

[2] You can check the result of a request by retrieving the system item **ReturnMessage** with DdeRequest.

D.5 GetList Command

Syntax

GetList(*listname*)

Description

With this command you read the **definition** of the list of variables called *listname* from the Server.

Lists of variables are named and defined with the DefineList command.

The GetList command can be executed only with the DDERequest function call.

Argument

Argument	Description
listname	Name of the list of variables whose definition is required to be read

Return values

DDEML Call	Variable	Value
DdeExecute	rw_bool	–
DDERequest	rw_int	No. of bytes
	rw_buffer	Names of the items in the list of variables
DdeAdvise	rw_bool	–
DdePoke	rw_bool	–

D.6 VarList Command

Syntax

VarList(*listname*, *var_list*)

Description

You can use this command for one-shot access using a list of variables (for DDE Request, DDE Advise, and DDE Poke).

The list of variables is not stored.

You do not need to define a list of variables for this and delete it again.

With write, an access status is not supplied for every variable.

The name of a variable may be defined on the symbol file of the DDE Server or specified as an absolute address.

A list of variables may contain up to 64 variables.

Argument

Argument	Description
listname	Name of the list of variables. The name is freely selectable. It is not evaluated by the DDE Server.
var_list	List of variables *var_list* is a string containing variables separated by a comma.

Return values

DDEML Call	Variable	Value
DdeExecute	rw_bool	True/False
DDERequest	rw_int	No. of bytes
	rw_buffer	Values of variables in CF_TEXT or CF_BYTESTREAM
DdeAdvise	rw_bool	True/False
DdePoke	rw_bool	True/False

Errors

Incorrect specification of variables does not result in rejection of the list. Instead, the access status of the different variables is set accordingly.

Errors are not supplied with write using VarList.

D.7 Result Command

Syntax

Result (*listname*)

Description

With this command you can read out the state of the last write access for the list of variables.

Access states are not deleted by reading. You can therefore fetch the result as often as you wish with Result.

Note

Warm links, hot links or pokes are not possible with this command.

The **Result** command can be executed only with the **DDERequest** function call.

Argument

Argument	Description
listname	Name of the list of variables whose write access state is required.

Return values

DDEML Call	Variable	Value
DdeExecute	rw_bool	–
DDERequest	rw_int	No. of bytes
	rw_buffer	Write state in CF_TEXT or CF_BYTESTREAM
DdeAdvise	rw_bool	–
DdePoke	rw_bool	–

Bibliography

E

/1/ SICOMP S5 SlotPLC
User’s Guide
Order No. 6ZB5 130 – 0BS02 – 0BAO

/2/ SIMATIC S5, COM 525 Communications Processors CP 524 and CP 525
Manual Vol. 2/2
Order No. 6ES5 998 – 1DB21

/3/ Microsoft E×cel
User’s Guide

/4/ MS Windows Software Development Kit
Programmer’s Reference
Volume 1: Overview
Microsoft Corporation

Index