

Control III Programming in C (small PLC)

Description of the commands

Subject to modifications without notice.

Generally, this manual refers to products without mentioning existing patents, utility models, or trademarks.

The absence of any such references does not indicate that a product is patent-free.

Table of Contents

Control III Programming in C (small PLC)

| | | |
|----------|--|----|
| 1 | Symbol Catalog | 7 |
| 2 | General | 8 |
| 2.1 | Product information | 8 |
| 2.2 | Introduction | 8 |
| 3 | Safety | 9 |
| 3.1 | Intended use | 9 |
| 3.2 | General safety information | 9 |
| 3.3 | Disposal | 9 |
| 4 | Programming Control III | 10 |
| 4.1 | Overview | 10 |
| 4.1.1 | ASi Data Exchange | 12 |
| 4.1.2 | AS-i Read IDI | 14 |
| 4.1.3 | AS-i Write ODI | 14 |
| 4.1.4 | AS-i Read ODI | 14 |
| 4.1.5 | AS-i Write Permanent Parameter | 15 |
| 4.1.6 | AS-i Read Permanent Parameter | 15 |
| 4.1.7 | AS-i Send Parameter | 15 |
| 4.1.8 | AS-i Read PI | 16 |
| 4.1.9 | AS-i Store PI | 16 |
| 4.1.10 | AS-i Read Duplicate Address List | 16 |
| 4.1.11 | AS-i Read Fault Detector | 17 |
| 4.1.12 | AS-i Write PCD | 17 |
| 4.1.13 | AS-i Read PCD | 17 |
| 4.1.14 | AS-i Store CDI | 18 |
| 4.1.15 | AS-i Read CDI | 18 |
| 4.1.16 | AS-i Write Extended ID1 | 18 |
| 4.1.17 | AS-i Write LPS | 19 |
| 4.1.18 | AS-i Read LPS | 20 |
| 4.1.19 | AS-i Read LAS | 20 |
| 4.1.20 | AS-i Read LDS | 21 |
| 4.1.21 | AS-i Read LCS | 21 |
| 4.1.22 | AS-i Write LOS | 21 |
| 4.1.23 | AS-i Read LOS | 22 |
| 4.1.24 | AS-i Read LPF | 22 |
| 4.1.25 | AS-i Read Ec Flags | 22 |
| 4.1.26 | AS-i set Config Mode | 23 |
| 4.1.27 | AS-i Write Hi Flags | 23 |
| 4.1.27.1 | AS-i Read Hi-Flags | 23 |
| 4.1.28 | AS-i Address Slave | 24 |
| 4.1.29 | AS-i Execute Command | 24 |

| | | |
|----------|--------------------------------------|-----------|
| 4.1.30 | AS-i Read All Config | 24 |
| 4.1.31 | AS-i Write All Config | 25 |
| 4.1.32 | AS-i read Error Counter | 25 |
| 4.1.33 | AS-i Mail Box | 25 |
| 4.1.34 | AS-i Write 16-bit ODI | 26 |
| 4.1.35 | AS-i Read 16Bit ODI | 26 |
| 4.1.36 | AS-i Read 16Bit IDI | 27 |
| 4.1.37 | AS-i Read Ctrl Acc ODI | 27 |
| 4.1.38 | AS-i Write Ctrl Acc ODI | 28 |
| 4.1.39 | AS-i Read Ctrl Acc AODI | 28 |
| 4.1.40 | AS-i Write Ctrl Acc AODI | 29 |
| 4.1.41 | Ctrl Init Timer | 29 |
| 4.1.42 | Ctrl Delay | 29 |
| 4.1.43 | Ctrl Init wgd | 29 |
| 4.1.44 | Ctrl Trigger wdg | 30 |
| 4.1.45 | Ctrl Eval Cycle time | 30 |
| 4.1.46 | Ctrl Read Parameter | 30 |
| 4.1.47 | Ctrl Write Parameter | 30 |
| 4.1.48 | Ctrl Read Flags | 31 |
| 4.1.49 | Ctrl Write Flags | 31 |
| 4.1.50 | Ctrl Read Key | 31 |
| 4.1.51 | Ctrl printf | 31 |
| 4.1.52 | Ctrl Breakpoint | 31 |
| 4.1.53 | Ctrl Display | 32 |
| 5 | Getting Started | 33 |
| 5.1 | Installing Eclipse Control III | 33 |
| 5.2 | Enabling Control III | 33 |
| 6 | Using Eclipse..... | 34 |
| 6.1 | Starting | 34 |
| 6.2 | Eclipse overview | 34 |
| 6.2.1 | The Project Explorer | 34 |
| 6.2.2 | Toolbar | 35 |
| 6.2.2.1 | Compiler | 35 |
| 6.2.2.2 | Debugging | 35 |
| 6.2.2.3 | Configuration tools | 35 |
| 6.2.3 | Editor | 36 |
| 6.2.4 | Console | 36 |
| 6.3 | File information | 37 |
| 6.4 | Setting the port | 38 |
| 6.5 | Creating a new project | 38 |
| 6.6 | A sample project | 39 |
| 6.6.1 | The C-code | 39 |
| 6.6.2 | Compiling | 42 |
| 6.6.3 | Downloading | 42 |
| 6.6.4 | Starting Control III | 42 |
| 6.7 | The Debugger | 42 |
| 6.7.1 | Initializing | 42 |

| | | |
|---------|--|----|
| 6.7.2 | Debugger overview | 43 |
| 6.7.2.1 | The Control Panel | 44 |
| 6.7.2.2 | Tasks | 44 |
| 6.7.2.3 | Debugger Overview | 44 |
| 6.7.2.4 | Code overview | 45 |
| 6.7.2.5 | Disassembly | 45 |
| 6.7.2.6 | Console | 45 |
| 6.7.3 | Start the Debugger | 45 |
| 6.7.4 | Example | 46 |
| 6.7.4.1 | Starting the Debugger | 50 |
| 6.7.4.2 | Using the Debugger | 51 |
| | | |
| 7 | Technical Data | 53 |
| 7.1 | Overview | 53 |
| 7.2 | Flags | 53 |
| 7.3 | Non-volatile parameters | 53 |
| 7.4 | Access rights to the output data area | 54 |
| | | |
| 8 | Error Messages | 55 |
| 8.1 | error: control not activated! | 55 |
| 8.2 | error: wrong control version | 55 |
| 8.3 | Launching problem | 55 |
| 8.4 | No or an incorrect cycle time is displayed. | 56 |
| 8.5 | The gateway doesn't stop at a breakpoint. | 56 |
| 8.6 | The program always goes to the same breakpoint. | 56 |
| 8.7 | The Control III can't affect any outputs. | 56 |

List of Figures

| | |
|--|----|
| Fig. 1. Eclipse Control III start window | 34 |
| Fig. 2. Eclipse settings.mak | 38 |
| Fig. 3. New Control III project | 39 |
| Fig. 4. Debug Configuration... .. | 46 |
| Fig. 5. Marking a breakpoint in the program code | 50 |
| Fig. 6. First breakpoint in Debug-mode | 51 |
| Fig. 7. Second breakpoint in Debug-mode | 51 |
| Fig. 8. Flag area | 53 |
| Fig. 9. Non-volatile parameters | 54 |
| Fig. 10. Access rights for the output data area | 54 |
| Fig. 11. error: control not activated | 55 |
| Fig. 12. error: wrong control version! | 55 |
| Fig. 13. Launching problem | 56 |

1. Symbol Catalog



Information!

This symbol indicates important information.

2. General

2.1 Product information

Control III performs all the tasks of a powerful small PLC and allows you to preprocess all sensor and actuator information.

2.2 Introduction

Control III, the AS-i Master PLC functionality integrated into the AS-i Master works together with standard AS-i I/O modules to form a small PLC with up to 248 in- and outputs per AS-i circuit, or up to 496 I/Os in a dual master. Standard C is used for all programming tasks of the small PLC.

Use of **Control III** in the gateways makes it possible to preprocess the sensor/actuator information. Typical applications include preprocessing of data as well as rapid execution of time-critical operations directly in the gateway.

The program for **Control III** is created on a standard PC and then uploaded to the AS-i master. A specially created Eclipse version (IDE) is provided as the programming tool. This includes all the tools needed for creating and testing the **Control III** program.

Control III is written in the widely used and very powerful "C" programming language. Various library functions serve to assist you and make programming easier.

The **Control III** program is loaded through one of the ports on the gateway and stored non-volatile in flash memory.

3 Safety

3.1 Intended use



Warning!

This symbol warns of a possible danger. The protection of operating personnel and the system against possible danger is not guaranteed if the control interface unit is not operated in accordance to its intended use.

3.2 General safety information



Warning!

Safety and correct functioning of the device cannot be guaranteed if any operation other than described in this operation manual is performed. Connecting the equipment and conducting any maintenance work under power must exclusively be performed by appropriately qualified personnel. In case a failure cannot be eliminated, the device must be taken out of operation and inadvertently operation must be prevented. Repair work must be performed by the manufacturer only. Additions or modifications to the equipment are not permitted and will void the warranty.



Information!

The operator is responsible for the observation of local safety standards.

3.3 Disposal



Information!

Electronic waste is hazardous waste. Please comply with all local ordinances when disposing this product!

The device does not contain batteries that need to be removed before disposing it.

4 Programming Control III

Programming of **Control III** is based on the widely-used and highly modular programming language 'C'. To create a Control III program, all the tasks that the gateway must assume must be written in 'C' and loaded into the gateway.

To program with **Control III**, the header file 'control.h' contains a series of library functions which can be used for running certain master functions. These functions are listed and explained in the following.

4.1 Overview

| | |
|---------------------------------|--|
| AASiDataExchange | AS-i Data Exchange is used for exchanging AS-i data and reads the 'execution control' flags. |
| AASiReadIDI | Reads the slave input data and the 'execution control' flags. |
| AASiWriteODI | Writes the slave output data. |
| AASiReadODI | Reads the output data from the AS-i master. |
| AASiWritePP | Writes the permanent parameters of a slave in the master. |
| AASiReadPP | Reads the permanent parameters of a slave in the master. |
| AASiSendParameter | Sends the parameters to a slave. |
| AASiReadPI | Reads the current parameters of a slave |
| AASiStorePI | Stores the current slave parameters as permanent parameters. |
| AASiReadDuplicateAdrList | Reads the list of all duplicate addresses. |
| AASiReadFaultDetector | Reads overvoltage, noise, EFLT and duplicate addresses. |
| AASiWritePCD | Writes the projected configuration of a slave. |
| AASiReadPCD | Reads the projected configuration of a slave. |
| AASiStoreCDI | Stores the current configuration as a permanent configuration. |
| AASiReadCDI | Reads the current configuration of a slave. |
| AASiWriteExtID1 | Writes the extended ID code 1 of Slave 0. |
| AASiWriteLPS | Writes the projected slave. |
| AASiReadLPS | Reads the projected slaves. |
| AASiReadLAS | Reads the activated slaves |
| AASiReadLDS | Reads the detected slaves. |
| AASiReadLCS | Reads all defective slaves. |
| AASiWriteLOS | List of the slaves which should go offline when there is a configuration error. |
| AASiReadLOS | Reads the list of the slaves which should go offline when there is a configuration error. |
| AASiReadLPF | Reads the list of slaves having a peripheral error. |

| | |
|------------------------------|--|
| AASiReadEcFlags | Reads the 'execution control' flags |
| AASiSetConfigMode | Sets the AS-i master to Configuration mode or to protected operating mode. |
| AASiWriteHiFlags | Writes the Host-Interface-Flags of the AS-i Master. |
| AASiReadHiFlags | Reads the Host-Interface-Flags of the AS-i Master. |
| AASiAddressSlave | Changes the address of a slave. |
| AASiExecuteCommand | AS-Interface command to be sent directly. |
| AASiReadAllConfig | Reads all configuration data (e.g. LPS, PP[] and PCD []) for all connected slaves. |
| AASiWriteAllConfig | Writes all configuration data (e.g. LPS, PP[] and PCD []) for all connected slaves. |
| AASiReadErrorCounters | Reads the Slave-Error-Counter |
| AASiMailbox | Generic Mailbox function. |
| AASiWrite16BitODI | Writes four channels of 16-bit ODI to an AS-i-Slave having e.g. Analog-Slave profile 7.3 or 7.4. |
| AASiRead16BitODI | Reads four channels of 16-bit ODI to an AS-i-Slave having e.g. Analog-Slave profile 7.3 or 7.4. |
| AASiRead16BitIDI | Reads four 16-bit IDI channels of an AS-i slave having e.g. Analog-Slave profile 7.3 or 7.4. |
| AASiReadCtrlAccODI | Reads the Control III authorization in order to change output data. |
| AASiWriteCtrlAccODI | Writes the Control III authorizations of the slave in order to change output data. |
| AASiReadCtrlAccAODI | Reads the Control III authorization in order to change analog output data. |
| AASiWriteCtrlAccAODI | Writes the Control III authorizations of the slave in order to change analog output data. |
| CCtrlInitTimer | Initialization of a timer function |
| CCtrlDelay | Delay in ms |
| CCtrlInitWdg | Initialization of the Control III watchdog |
| CCtrlTriggerWdg | Trigger Control III Watchdog. |
| CCtrlEvalCycletime | Evaluates the Control III cycle time. |
| CCtrlReadParameter | Reads Control NV-Parameter. |
| CCtrlWriteParameter | Writes Control NV-Parameter. |
| CCtrlReadFlags | Reads Control-Flags. |
| CCtrlWriteFlags | Writes Control-Flags. |
| CCtrlReadkey | Reads user-defined unique control key. |
| CCtrlPrintf | Printf function |
| CCtrlBreakpoint | Initializes the Debugger |
| CCtrlDisplay | Control III display function |

4.1.1 ASi Data Exchange

AS-i Data Exchange is used for exchanging AS-i data between the master and the application and reads the 'execution control' flags.

```
int (*AASiDataExchange) (unsigned char Circuit, AASiProcessData ODI, AASiProcessData IDI, AASiEcFlags *EcFlags);
```

Parameter:

Circuit: AS-i Master circuit
 ODI: 32 bytes output data

Return:

IDI: 32 bytes input data
 EcFlags: Execution control flags (2 Bytes) of the AS-i Master.

AS-i circuit 1 / 2 : Input data image IDI

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|--------------|----|----|--------------|----|----|----|----|--------------|----|----|--------------|----|----|----|----|----|
| Byte | D3 | D2 | D1 | D0 | D3 | D2 | D1 | D0 | Byte | D3 | D2 | D1 | D0 | D3 | D2 | D1 | D0 |
| 0 | Slave 1/1A | | | Slave 0/0A | | | | 1 | Slave 3/3A | | | Slave 2/2A | | | | | |
| 2 | Slave 5/5A | | | Slave 4/4A | | | | 3 | Slave 7/7A | | | Slave 6/6A | | | | | |
| 4 | Slave 9/9A | | | Slave 8/8A | | | | 5 | Slave 11/11A | | | Slave 10/10A | | | | | |
| 6 | Slave 13/13A | | | Slave 12/12A | | | | 7 | Slave 15/15A | | | Slave 14/14A | | | | | |
| 8 | Slave 17/17A | | | Slave 16/16A | | | | 9 | Slave 19/19A | | | Slave 18/18A | | | | | |
| 10 | Slave 21/21A | | | Slave 20/20A | | | | 11 | Slave 23/23A | | | Slave 22/22A | | | | | |
| 12 | Slave 25/25A | | | Slave 24/24A | | | | 13 | Slave 27/27A | | | Slave 26/26A | | | | | |
| 14 | Slave 29/29A | | | Slave 28/28A | | | | 15 | Slave 31/31A | | | Slave 30/30A | | | | | |
| 16 | Slave 1/1B | | | Slave 0/0B | | | | 17 | Slave 3/3B | | | Slave 2/2B | | | | | |
| 18 | Slave 5/5B | | | Slave 4/4B | | | | 19 | Slave 7/7B | | | Slave 6/6B | | | | | |
| 20 | Slave 9/9B | | | Slave 8/8B | | | | 21 | Slave 11/11B | | | Slave 10/10B | | | | | |
| 22 | Slave 13/13B | | | Slave 12/12B | | | | 23 | Slave 15/15B | | | Slave 14/14B | | | | | |
| 24 | Slave 17/17B | | | Slave 16/16B | | | | 25 | Slave 19/19B | | | Slave 18/18B | | | | | |
| 26 | Slave 21/21B | | | Slave 20/20B | | | | 27 | Slave 23/23B | | | Slave 22/22B | | | | | |
| 28 | Slave 25/25B | | | Slave 24/24B | | | | 29 | Slave 27/27B | | | Slave 26/26B | | | | | |
| 30 | Slave 29/29B | | | Slave 28/28B | | | | 31 | Slave 31/31B | | | Slave 30/30B | | | | | |

Tab. 4-1. Input data image IDI

AS-i circuit 1 / 2 : Output data image ODI

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|--------------|----|----|--------------|----|----|----|----|--------------|----|----|--------------|----|----|----|----|----|
| Byte | D3 | D2 | D1 | D0 | D3 | D2 | D1 | D0 | Byte | D3 | D2 | D1 | D0 | D3 | D2 | D1 | D0 |
| 0 | Slave 1/1A | | | Slave 0/0A | | | | 1 | Slave 3/3A | | | Slave 2/2A | | | | | |
| 2 | Slave 5/5A | | | Slave 4/4A | | | | 3 | Slave 7/7A | | | Slave 6/6A | | | | | |
| 4 | Slave 9/9A | | | Slave 8/8A | | | | 5 | Slave 11/11A | | | Slave 10/10A | | | | | |
| 6 | Slave 13/13A | | | Slave 12/12A | | | | 7 | Slave 15/15A | | | Slave 14/14A | | | | | |
| 8 | Slave 17/17A | | | Slave 16/16A | | | | 9 | Slave 19/19A | | | Slave 18/18A | | | | | |
| 10 | Slave 21/21A | | | Slave 20/20A | | | | 11 | Slave 23/23A | | | Slave 22/22A | | | | | |
| 12 | Slave 25/25A | | | Slave 24/24A | | | | 13 | Slave 27/27A | | | Slave 26/26A | | | | | |
| 14 | Slave 29/29A | | | Slave 28/28A | | | | 15 | Slave 31/31A | | | Slave 30/30A | | | | | |
| 16 | Slave 1/1B | | | Slave 0/0B | | | | 17 | Slave 3/3B | | | Slave 2/2B | | | | | |
| 18 | Slave 5/5B | | | Slave 4/4B | | | | 19 | Slave 7/7B | | | Slave 6/6B | | | | | |
| 20 | Slave 9/9B | | | Slave 8/8B | | | | 21 | Slave 11/11B | | | Slave 10/10B | | | | | |
| 22 | Slave 13/13B | | | Slave 12/12B | | | | 23 | Slave 15/15B | | | Slave 14/14B | | | | | |
| 24 | Slave 17/17B | | | Slave 16/16B | | | | 25 | Slave 19/19B | | | Slave 18/18B | | | | | |
| 26 | Slave 21/21B | | | Slave 20/20B | | | | 27 | Slave 23/23B | | | Slave 22/22B | | | | | |
| 28 | Slave 25/25B | | | Slave 24/24B | | | | 29 | Slave 27/27B | | | Slave 26/26B | | | | | |
| 30 | Slave 29/29B | | | Slave 28/28B | | | | 31 | Slave 31/31B | | | Slave 30/30B | | | | | |

Tab. 4-2. Output data image ODI

4.1.2 AS-i Read IDI

AS-i Read IDI reads the input data from slaves and the execution control flags.

```
int (*AASiReadIDI) (unsigned char Circuit, AASiProcessData IDI,  
AASiSlaveAddr First, unsigned char Amount, AASiEcFlags  
*EcFlags);
```

Parameter:

Circuit: AS-i Master circuit
First: Index of the first slaves
Amount: Number of slaves to read

Return:

IDI: 32 bytes input data
Each slave uses 4 bits (nibble) of a byte. Unused bytes are set to null

EcFlags: Execution control flags (2 Bytes) of the AS-i master

4.1.3 AS-i Write ODI

AS-i Write ODI writes the output data from slaves.

```
int (*AASiWriteODI) (unsigned char Circuit, AASiProcessData ODI,  
AASiSlaveAddr First, unsigned char Amount);
```

Parameter:

Circuit: AS-i Master circuit
ODI: 32 bytes output data
First: Index of the first slave
Amount: Number of slaves to be written

Each slave uses 4 bits (nibble) of a byte. Unused bytes are set to null.

Return: —

4.1.4 AS-i Read ODI

AS-i Read ODI reads the output data from the AS-i Master.

```
int (*AASiReadODI) (unsigned char Circuit, AASiProcessData ODI);
```

Parameter:

Circuit AS-i Master circuit

Return:

ODI: 32 bytes output data
Each slave uses 4 bits (nibble) of a byte. Unused bytes are set to null.

4.1.5 AS-i Write Permanent Parameter

AS-i Write Permanent Parameter writes the permanent parameters of a slave in the Master.

```
int (*AASiWritePP) (unsigned char Circuit, AASiSlaveAddr  
Address, AASiSlaveData PP);
```

Parameter:

Circuit: AS-i Master circuit
Address: Slave address
PP: Permanent Parameters (low Nibble)

Return: —

4.1.6 AS-i Read Permanent Parameter

AS-i Read Permanent Parameter reads the permanent parameters of a slave in the Master.

```
int (*AASiReadPP) (unsigned char Circuit, AASiSlaveAddr Address,  
AASiSlaveData *PP);
```

Parameter

Circuit: AS-i Master circuit
Address: Slave address

Return:

PP: Permanent parameters (low nibble)

4.1.7 AS-i Send Parameter

AS-i Send Parameter sends the parameters to a slave.

```
int (*AASiSendParameter) (unsigned char Circuit, AASiSlaveAddr  
Address, AASiSlaveData PI, AASiSlaveData *Return);
```

Parameter:

Circuit: AS-i Master circuit
Address: Slave address
PI: Parameters to be sent (low Nibble)

Return:

Return: In case of error PI is returned inverted. .

4.1.8 AS-i Read PI

AS-i Read PI reads the current parameters of a slave.

```
int (*AASiReadPI) (unsigned char Circuit, AASiSlaveAddr Address,  
AASiSlaveData *PI);
```

Parameter:

Circuit: AS-i Master circuit
Address: Slave address

Return:

PI: Parameters to be sent (low Nibble)

4.1.9 AS-i Store PI

AS-i Store PI stores the current slave parameters as permanent parameters.

```
int (*AASiStorePI) (unsigned char Circuit);
```

Parameter:

Circuit: AS-i Master circuit
Address: Slave address

Return: —

4.1.10 AS-i Read Duplicate Address List

AS-i Read Duplicate Address List reads the list of all duplicate addresses.

```
int (*AASiReadDuplicateAdrList) (unsigned char Circuit,  
AASiSlaveList, *DpAdrList);
```

Parameter:

Circuit: AS-i Master circuit

Return:

DpAdrList: List of duplicate addresses

4.1.11 AS-i Read Fault Detector

AS-i Read Fault Detector reads overvoltage, noise, EFLT and duplicate addresses.

```
int (*AASiReadFaultDetector) (unsigned char Circuit,
    unsigned char *pucFaultDetectorActiv,
    unsigned char *pucFaultDetectorHistoric);
```

Parameter:

Circuit: AS-i Master circuit

Return:

PucFaultDetectorActiv: Active fault detector
PucFaultDetectorHistoric: Historic fault detector

4.1.12 AS-i Write PCD

AS-i Write PCD writes the projected configuration of a slave.

```
int (*AASiWritePCD) (unsigned char Circuit, AASiSlaveAddr
    Address, AASiConfigData PCD);
```

Parameter:

Circuit: AS-i Master circuit
Address: Slave address

Return:

PCD: Projected slave configuration to be written

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|--------------|----|----|----|-------------------|----|----|----|
| Byte | D3 | D2 | D1 | D0 | D3 | D2 | D1 | D0 |
| 0 | Slave ID | | | | I/O Configuration | | | |
| 1 | Extended ID2 | | | | Extended ID1 | | | |

Tab. 4-3. PCD configuration

4.1.13 AS-i Read PCD

AS-i Read PCD reads the projected configuration of a slave.

```
int (*AASiReadPCD) (unsigned char Circuit, AASiSlaveAddr
    Address, AASiConfigData *PCD);
```

Parameter:

Circuit: AS-i Master circuit
Address: Slave address

Return:

PCD: Projected Slave configuration (see Tab. <PCD configuration>).

4.1.14 AS-i Store CDI

AS-i Store CDI stores the current configuration as a permanent configuration.

```
int (*AASiStoreCDI) (unsigned char Circuit);
```

Parameter:

Circuit: AS-i Master circuit

4.1.15 AS-i Read CDI

AS-i Read CDI reads the current configuration of a slave.

```
int (*AASiReadCDI) (unsigned char Circuit, AASiSlaveAddr Address, AASiConfigData *CDI);
```

Parameter:

Circuit: AS-i Master circuit

Address: Slave address

Return:

CDI: Slave configuration

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|--------------|----|----|----|-------------------|----|----|----|
| Byte | D3 | D2 | D1 | D0 | D3 | D2 | D1 | D0 |
| 0 | Slave ID | | | | I/O Configuration | | | |
| 1 | Extended ID2 | | | | Extended ID1 | | | |

Tab. 4-4. CDI configuration

4.1.16 AS-i Write Extended ID1

AS-i Write Extended ID1 writes the extended ID-Code 1 von Slave 0.

```
int (*AASiWriteExtID1) (unsigned char Circuit, AASiSlaveData ID1);
```

Parameter:

Circuit: AS-i Master circuit

Address: Slave address

ID1: Extended ID-Code 1

Return:

Special error codes which describe the reason for the faulty transmission.

4.1.17 AS-i Write LPS

AS-i Write LPS writes the projected slaves.

```
int (*AASiWriteLPS) (unsigned char Circuit, AASiSlaveList LPS);
```

Parameter:

Circuit: AS-i Master circuit

LPS: 8-byte Slave list

Each bit in the LPS corresponds to a slave as follows:

| Bit | Slave | Bit | Slave | Bit | Slave | Bit | Slave |
|-----|------------------------------|-----|--------------|-----|------------------------------|-----|--------------|
| 0 | Slave 0/0A cannot be set! | 16 | Slave 16/16A | 32 | Slave 0/0B cannot be set! | 48 | Slave 16/16B |
| 1 | Slave 1/1A | 17 | Slave 17/17A | 33 | Slave 1/1B | 49 | Slave 17/17B |
| 2 | Slave 2/2A | 18 | Slave 18/18A | 34 | Slave 2/2B | 50 | Slave 18/18B |
| 3 | Slave 3/3A | 19 | Slave 19/19A | 35 | Slave 3/3B | 51 | Slave 19/19B |
| 4 | Slave 4/4A | 20 | Slave 20/20A | 36 | Slave 4/4B | 52 | Slave 20/20B |
| 5 | Slave 5/5A | 21 | Slave 21/21A | 37 | Slave 5/5B | 53 | Slave 21/21B |
| 6 | Slave 6/6A | 22 | Slave 22/22A | 38 | Slave 6/6B | 54 | Slave 22/22B |
| 7 | Slave 7/7A | 23 | Slave 23/23A | 39 | Slave 7/7B | 55 | Slave 23/23B |
| 8 | Slave 8/8A | 24 | Slave 24/24A | 40 | Slave 8/8B | 56 | Slave 24/24B |
| 9 | Slave 9/9A | 25 | Slave 25/25A | 41 | Slave 9/9B | 57 | Slave 25/25B |
| 10 | Slave 10/10A | 26 | Slave 26/26A | 42 | Slave 10/10B | 58 | Slave 26/26B |
| 11 | Slave 11/11A | 27 | Slave 27/27A | 43 | Slave 11/11B | 59 | Slave 27/27B |
| 12 | Slave 12/12A | 28 | Slave 28/28A | 44 | Slave 12/12B | 60 | Slave 28/28B |
| 13 | Slave 13/13A | 29 | Slave 29/29A | 45 | Slave 13/13B | 61 | Slave 29/29B |
| 14 | Slave 14/14A | 30 | Slave 30/30A | 46 | Slave 14/14B | 62 | Slave 30/30B |
| 15 | Slave 15/15A | 31 | Slave 31/31A | 47 | Slave 15/15B | 63 | Slave 31/31B |

Tab. 4-5. LPS

The slave is projected if the bit is set.

4.1.18 AS-i Read LPS

AS-i Read LPS reads the projected slaves.

```
int (*AASiReadLPS) (unsigned char Circuit, AASiSlaveList *LPS);
```

Parameter:

Circuit: AS-i Master circuit

Return:

LPS: 8-byte slave list (see Tab. <LPS>)

The slave is projected if the bit is set.

4.1.19 AS-i Read LAS

AS-i Read LAS reads the activated slaves.

```
int (*AASiReadLAS) (unsigned char Circuit, AASiSlaveList *LAS);
```

Parameter:

Circuit: AS-i Master circuit

LAS: 8-byte slave list

Each bit in the LAS corresponds to a slave.

| Bit | Slave | Bit | Slave | Bit | Slave | Bit | Slave |
|-----|--------------|-----|--------------|-----|--------------|-----|--------------|
| 0 | Slave 0/0A | 16 | Slave 16/16A | 32 | Slave 0/0B | 48 | Slave 16/16B |
| 1 | Slave 1/1A | 17 | Slave 17/17A | 33 | Slave 1/1B | 49 | Slave 17/17B |
| 2 | Slave 2/2A | 18 | Slave 18/18A | 34 | Slave 2/2B | 50 | Slave 18/18B |
| 3 | Slave 3/3A | 19 | Slave 19/19A | 35 | Slave 3/3B | 51 | Slave 19/19B |
| 4 | Slave 4/4A | 20 | Slave 20/20A | 36 | Slave 4/4B | 52 | Slave 20/20B |
| 5 | Slave 5/5A | 21 | Slave 21/21A | 37 | Slave 5/5B | 53 | Slave 21/21B |
| 6 | Slave 6/6A | 22 | Slave 22/22A | 38 | Slave 6/6B | 54 | Slave 22/22B |
| 7 | Slave 7/7A | 23 | Slave 23/23A | 39 | Slave 7/7B | 55 | Slave 23/23B |
| 8 | Slave 8/8A | 24 | Slave 24/24A | 40 | Slave 8/8B | 56 | Slave 24/24B |
| 9 | Slave 9/9A | 25 | Slave 25/25A | 41 | Slave 9/9B | 57 | Slave 25/25B |
| 10 | Slave 10/10A | 26 | Slave 26/26A | 42 | Slave 10/10B | 58 | Slave 26/26B |
| 11 | Slave 11/11A | 27 | Slave 27/27A | 43 | Slave 11/11B | 59 | Slave 27/27B |
| 12 | Slave 12/12A | 28 | Slave 28/28A | 44 | Slave 12/12B | 60 | Slave 28/28B |
| 13 | Slave 13/13A | 29 | Slave 29/29A | 45 | Slave 13/13B | 61 | Slave 29/29B |
| 14 | Slave 14/14A | 30 | Slave 30/30A | 46 | Slave 14/14B | 62 | Slave 30/30B |
| 15 | Slave 15/15A | 31 | Slave 31/31A | 47 | Slave 15/15B | 63 | Slave 31/31B |

Tab. 4-6. LAS, LDS, LOS and LPF

The slave is activated if the bit is set.

4.1.20 AS-i Read LDS

AS-i Read LDS reads the detected slaves laves.

```
int (*AASiReadLDS) (unsigned char Circuit, AASiSlaveList *LDS);
```

Parameter:

Circuit: AS-i Master circuit

Return:

LDS: 8-byte slave list

Each bit in the LDS corresponds to a slave (see Tab. <LAS, LDS, LOS and LPF>).

4.1.21 AS-i Read LCS

AS-i Read LCS reads all faulty slaves.

```
int (*AASiReadLCS) (unsigned char Circuit, AASiSlaveList *LCS);
```

Parameter:

Circuit: AS-i Master circuit

Return:

LCS: 8-byte slave list

Each bit in the LCS corresponds to a slave (see Tab. <LAS, LDS, LOS and LPF>).



Information!

The list is reset after reading!

4.1.22 AS-i Write LOS

AS-i Write LOS writes the list of those slaves which should go offline when there is a configuration error.

```
int (*AASiWriteLOS) (unsigned char Circuit, AASiSlaveList LOS);
```

Parameter:

Circuit: AS-i Master circuit

Return:

LOS: 8-byte slave address list

Each bit in the LOS corresponds to a slave (see Tab. <LAS, LDS, LOS and LPF>).

4.1.23 AS-i Read LOS

AS-i Write LOS reads the list of those slaves which should go offline when there is a configuration error.

```
int (*AASiReadLOS) (unsigned char Circuit, AASiSlaveList *LOS);
```

Parameter:

Circuit: AS-i Master circuit

Return:

LOS: 8-byte slave address list

Each bit in the LOS corresponds to a slave (see Tab. <LAS, LDS, LOS and LPF>).

4.1.24 AS-i Read LPF

AS-i Read LPF reads the list of slaves with a peripheral error.

```
int (*AASiReadLPF) (unsigned char Circuit, AASiSlaveList *LPF);
```

Parameter:

Circuit: AS-i Master circuit

Return:

LPF: 8-byte slave address list

Each bit in the LPS corresponds to a slave (see Tab. <LAS, LDS, LOS and LPF>).

4.1.25 AS-i Read Ec Flags

AS-i Read Ec Flags reads the execution control flags.

```
int (*AASiReadEcFlags) (unsigned char Circuit, AASiEcFlags *EcFlags);
```

Parameter:

Circuit: AS-i Master circuit

Return:

EcFlags: Two-byte EcFlags.

4.1.26 AS-i set Config Mode

AS-i set Config Mode sets the AS-i Master to Configuration mode or to protected operating mode.

```
int (*AASiSetConfigMode) (unsigned char Circuit, unsigned char Mode);
```

Parameter:

Circuit: AS-i Master circuit
Mode: 1 = Configuration mode
0 = Protected operating mode

Return: —

4.1.27 AS-i Write Hi Flags

AS-i Write Hi Flags writes Host-Interface-Flags of the AS-i master.

```
int (*AASiWriteHiFlags) (unsigned char Circuit, AASiHiFlags HiFlags);
```

Parameter:

Circuit: AS-i Master circuit
HiFlags: Host-Interface-Flags (one byte)

Return: —

4.1.27.1 AS-i Read Hi-Flags

AS-i Read Hi-Flags reads the Host-Interface-Flags of the AS-i master.

```
int (*AASiReadHiFlags) (unsigned char Circuit, AASiHiFlags *HiFlags);
```

Parameter:

Circuit: AS-i Master circuit

Return:

HiFlags: Host-Interface-Flags (one byte)

4.1.28 AS-i Address Slave

AS-i Address Slave changes the address of a slave.

```
int (*AASiAddressSlave) (unsigned char Circuit, AASiSlaveAddr  
OldAddress, AASiSlaveAddr NewAddress);
```

Parameter:

Circuit: AS-i Master circuit
OldAddress: Old slave address
NewAddress: New slave address

Return:

Special error codes which describe the reason for the defective transmission.

4.1.29 AS-i Execute Command

AS-i Execute Command is an AS-i command to be sent directly.

```
int (*AASiExecuteCommand) (unsigned char Circuit, AASiSlaveAddr  
Address, AASiSlaveData Request, AASiSlaveData *Response);
```

Parameter:

Circuit: AS-i Master circuit
Address: Slave address
Request: AS-i Request command

Return:

Response: AS-i Request

4.1.30 AS-i Read All Config

AS-i Read All Config reads all configuration data (e.g. LPS, PP[] and PCD []) for all connected slaves.

```
int (*AASiReadAllConfig) (unsigned char Circuit, AASiConfig  
*Configurations );
```

Parameter:

Circuit: AS-i Master circuit
Configurations: Array for recording the configuration data

Return:

The configuration data in Configurations []

4.1.31 AS-i Write All Config

AS-i Write All Config writes all configuration data (e.g. LPS, PP[] and PCD []) for all connected slaves.

```
int (*AASiWriteAllConfig) (unsigned char Circuit, AASiConfig Configurations);
```

Parameter:

Circuit: AS-i Master circuit
Configurations: Array for recording the configuration data

Return: —

4.1.32 AS-i read Error Counter

AS-i read Error Counter reads the Slave-Error-Counter.

```
int (*AASiReadErrorCounters) (unsigned char Circuit, AASiErrorCounters Counters);
```

Parameter:

Circuit: AS-i Master circuit

Return:

Counters: 64 bytes (one byte per slave)



Information!

The list is reset after reading!

4.1.33 AS-i Mail Box

Generic Mailbox function

```
int (*AASiMailbox) (unsigned char Circuit, AASiMbRequestType Request, int ExpResLen, AASiMbResponseType *Response);
```

Parameter:

Circuit: AS-i Master circuit
Request: Structure for Mailbox query
ExpResLen: Expected length of the response (-1 = unknown)

Return:

Response: Structure for Mailbox response

4.1.34 AS-i Write 16-bit ODI

AS-i Write 16Bit ODI writes four 16-bit ODI channels of an AS-i slave having e.g. Analog-Slave-Profile 7.3 or 7.4.

```
int (*AASiWrite16BitODI) (unsigned char Circuit, AASiSlaveAddr  
Address, AASi16BitData Out);
```

Parameter:

Circuit: AS-i Master circuit
Address: Slave address
ExpResLen: Expected length of the response (-1 = unknown)
Out: 4 channels with 16-bit values
Word 0 : Channel 1
Word 1 : Channel 2
Word 2 : Channel 3
Word 3 : Channel 4

Return:—

4.1.35 AS-i Read 16Bit ODI

AS-i Read 16Bit ODI reads four 16-bit ODI channels of an AS-i slave having e.g. Analog-Slave-Profile 7.3 or 7.4.

```
int (*AASiRead16BitODI) (unsigned char Circuit, AASiSlaveAddr  
Address, AASi16BitData In);
```

Parameter:

Circuit: AS-i Master circuit
Address: Slave address
In: 4 channels with 16-bit values
Word 0 : Channel 1
Word 1 : Channel 2
Word 2 : Channel 3
Word 3 : Channel 4

4.1.36 AS-i Read 16Bit IDI

AS-i Read 16Bit IDI reads four 16-bit ODI channels of an AS-i slave having e.g. Analog-Slave-Profile 7.3 or 7.4.

```
int (*AASiRead16BitIDI) (unsigned char Circuit, unsigned char  
Address, AASi16BitData In);
```

Parameter:

Circuit: AS-i Master circuit
Address: Slave address
In: 4 channels with 16-bit values
Word 0 : Channel 1
Word 1 : Channel 2
Word 2 : Channel 3
Word 3 : Channel 4

4.1.37 AS-i Read Ctrl Acc ODI

AS-i Read Ctrl Acc ODI reads the Control III authorizations for writing slave output data.

```
int (*AASiReadCtrlAccODI) (unsigned char Circuit,  
AASiCtrlAccODI ODI );
```

Parameter:

Circuit: AS-i Master circuit

Return:

ODI: 32 bytes Ctrl access data

4.1.38 AS-i Write Ctrl Acc ODI

AS-i Write Ctrl Acc ODI writes the Control III slave authorizations for changing output data.

```
int (*AASiWriteCtrlAccODI) (unsigned char Circuit,
AASiCtrlAccODI ODI, AASiSlaveAddr First, unsigned char Amount);
```

Parameter:

- Circuit: AS-i Master circuit
- ODI: 32 bytes Ctrl access data
- First: Index of the first slave
- Amount: Number of slaves following the first

Return:—

4.1.39 AS-i Read Ctrl Acc AODI

AS-i Read Ctrl Acc AODI reads the Control III authorizations for writing analog slave output data.

```
int (*AASiReadCtrlAccAODI) (unsigned char Circuit,
AASiCtrlAccAODI AODI);
```

Parameter:

- Circuit: AS-i Master circuit

Return:

AODI: 16 bytes output data Ctrl access

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---------|--------------|----|----|--------------|----|----|----|----|--------------|----|----|--------------|----|----|----|----|----|
| Channel | 3 | 2 | 1 | 0 | 3 | 2 | 1 | 0 | Channel | 3 | 2 | 1 | 0 | 3 | 2 | 1 | 0 |
| Byte | D3 | D2 | D1 | D0 | D3 | D2 | D1 | D0 | Byte | D3 | D2 | D1 | D0 | D3 | D2 | D1 | D0 |
| 0 | Slave 1/1A | | | Slave 0/0A | | | | 1 | Slave 3/3A | | | Slave 2/2A | | | | | |
| 2 | Slave 5/5A | | | Slave 4/4A | | | | 3 | Slave 7/7A | | | Slave 6/6A | | | | | |
| 4 | Slave 9/9A | | | Slave 8/8A | | | | 5 | Slave 11/11A | | | Slave 10/10A | | | | | |
| 6 | Slave 13/13A | | | Slave 12/12A | | | | 7 | Slave 15/15A | | | Slave 14/14A | | | | | |
| 8 | Slave 17/17A | | | Slave 16/16A | | | | 9 | Slave 19/19A | | | Slave 18/18A | | | | | |
| 10 | Slave 21/21A | | | Slave 20/20A | | | | 11 | Slave 23/23A | | | Slave 22/22A | | | | | |
| 12 | Slave 25/25A | | | Slave 24/24A | | | | 13 | Slave 27/27A | | | Slave 26/26A | | | | | |
| 14 | Slave 29/29A | | | Slave 28/28A | | | | 15 | Slave 31/31A | | | Slave 30/30A | | | | | |
| 16 | Slave 1/1B | | | Slave 0/0B | | | | 17 | Slave 3/3B | | | Slave 2/2B | | | | | |
| 18 | Slave 5/5B | | | Slave 4/4B | | | | 19 | Slave 7/7B | | | Slave 6/6B | | | | | |
| 20 | Slave 9/9B | | | Slave 8/8B | | | | 21 | Slave 11/11B | | | Slave 10/10B | | | | | |
| 22 | Slave 13/13B | | | Slave 12/12B | | | | 23 | Slave 15/15B | | | Slave 14/14B | | | | | |
| 24 | Slave 17/17B | | | Slave 16/16B | | | | 25 | Slave 19/19B | | | Slave 18/18B | | | | | |
| 26 | Slave 21/21B | | | Slave 20/20B | | | | 27 | Slave 23/23B | | | Slave 22/22B | | | | | |
| 28 | Slave 25/25B | | | Slave 24/24B | | | | 29 | Slave 27/27B | | | Slave 26/26B | | | | | |
| 30 | Slave 29/29B | | | Slave 28/28B | | | | 31 | Slave 31/31B | | | Slave 30/30B | | | | | |

Tab. 4-7. Output data image AODI

4.1.40 AS-i Write Ctrl Acc AODI

AS-i Write Ctrl Acc AODI writes the Control III slave authorizations for changing analog output data.

```
int (*AASiWriteCtrlAccAODI)( unsigned char Circuit,
AASiCtrlAccAODI AODI, AASiSlaveAddr First, unsigned char Amount;
```

Parameter:

Circuit: AS-i Master circuit

AODI: 16 bytes Ctrl access data (see Tab. <Output data image AODI>)

First: Index of the first slave

Amount: Number of slaves following the first

Return:—

4.1.41 Ctrl Init Timer

Initialization of a Timer Interrupt

```
int (*CCtrlInitTimer) (unsigned long ticks_ms, void
(*timer_func)( void ) );
```

Parameter:

Circuit: AS-i Master circuit

ticks_ms: Interrupt Time in ms

timer_func: Callback function of the timer interrupt

Return:—

4.1.42 Ctrl Delay

Delay function

```
int (*CCtrlDelay) ( unsigned long ticks_ms );
```

Parameter:

Circuit: AS-i Master circuit

ticks_ms: Delay in ms

Return:—

4.1.43 Ctrl Init wgd

Initialization of a watchdog for Control III

```
int (*CCtrlInitWdg) ( unsigned long ticks );
```

Parameter:

Circuit: AS-i Master circuit

ticks: Watchdog time in ms

Return:—

4.1.44 Ctrl Trigger wdg

Triggers the Control III Watchdogs

```
int (*CctrlTriggerWdg) ( void );
```

Parameter: —

Return:—

4.1.45 Ctrl Eval Cycle time

The function determines the cycle time of the Control III program.

```
int (*CctrlEvalCycletime) ( void );
```

Parameter: —

Return:—

4.1.46 Ctrl Read Parameter

Ctrl Read Parameter reads non-volatile data from the flash memory.

```
int (*CctrlReadParameter) (unsigned char *buffer, unsigned short len, unsigned short adr);
```

Parameter:

len: Length of the memory to be read

adr: Address of the first byte in the data to be read

Return:

buffer: Read-buffer

4.1.47 Ctrl Write Parameter

Ctrl Write Parameter writes non-volatile data to the flash memory.

```
int (*CctrlWriteParameter) ( unsigned char *buffer, unsigned short len, unsigned short adr );
```

Parameter:

buffer: Write-buffer

len: Length of the buffer

adr: Address of the first byte to which the data should be written

Return:—

4.1.48 Ctrl Read Flags

Ctrl Read Flags reads the AS-i-Control-Flags.

```
int (*CctrlReadFlags) ( unsigned char *flags );
```

Parameter: —

Return:

flags AS-i-Control-Flags

4.1.49 Ctrl Write Flags

Ctrl Write Flags writes the AS-i-Control-Flags.

```
int (*CctrlWriteFlags) ( unsigned char flags );
```

Parameter:

flags: AS-i-Control-Flags

Return:—

4.1.50 Ctrl Read Key

Ctrl Read Key reads user-defined, unique Control III keys.

```
int (*CctrlReadKey) ( unsigned int *key );
```

Parameter: —

Return:

key: User-defined Control III key

4.1.51 Ctrl printf

Printf function

```
int (*CctrlPrintf) ( const char *format, ... );
```

Parameter: —

Return:—

4.1.52 Ctrl Breakpoint

Debugger initialization; "First" Breakpoint.

```
void (*CctrlBreakpoint) (void);
```

Parameter: —

Return:—

4.1.53 Ctrl Display

Self-defined display on the gateway display

```
int (*CctrlDisplay)( unsigned char mode, cctrl_disp_t  
disp_buffer );
```

Parameter:

```
mode CCTRL_DISP_MODE_TRADITIONAL  
CCTRL_DISP_MODE_SPONTANEOUS  
disp_buffer.show: 0 = Clear  
                  1 = Show  
disp_buffer.type: CCTRL_DISP_TYP_4LINES  
                  = 4 lines of text  
                  CCTRL_DISP_TYP_BIGNUM  
                  = Large number + 1 line of text  
disp_buffer.time: Time for showing on the Display (min  
                  2 sec.)  
disp_buffer.big:  Buffer for large number  
disp_buffer.lines: Buffer for 4 lines of text
```

Return: 0 = OK
!0 = not OK

5 Getting Started

Eclipse is a widely used and open-source programming tool for developing various kinds of software. Eclipse has been adapted for using **Control III** so that it assists the user in implementing his software program.



5.1 Installing Eclipse Control III

Run the 'setup.exe', select a destination directory and click on the 'Install' button. Installation may take several minutes.



Information!

No entries are made in the Windows Start menu. The files are simply extracted and saved to the corresponding directory.

5.2 Enabling Control III

If your gateway is not yet enabled for using **Control III**, you can request enabling.

If you already have an enabling code for your gateway, you will find an unlock control tool under the Eclipse configuration tools (see Sec. <Unlock Control:>).

Select your gateway and enter the corresponding enable code. The **Control III** function is enabled by clicking on 'Unlock/Lock'. After successful activation you are prompted to start the gateway. Additional information can be found in the document "Control_enabling.pdf".

6. Using Eclipse

After successful installation you may proceed directly creating your own Control III program.

6.1 Starting

Follow the installation path (default C:/Programs) to the "eclipse_control" folder and from there run "eclipse.exe".

6.2 Eclipse overview

The Eclipse start screen is clearly divided into the most important areas.

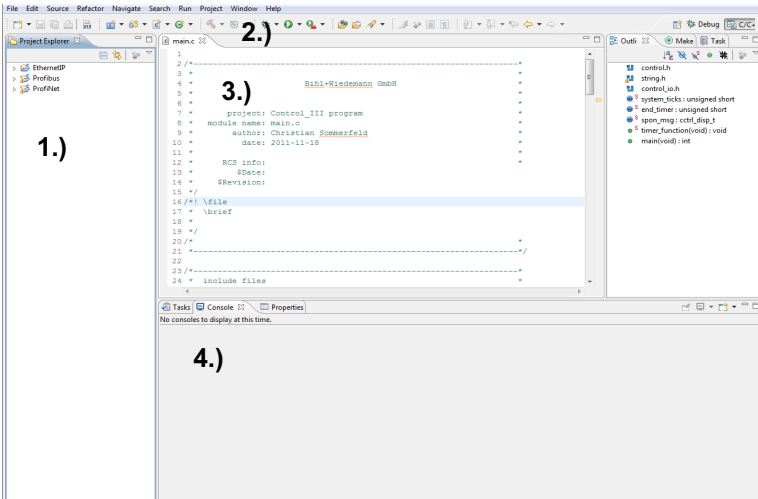


Fig. 6-1. Eclipse Control III start window

1. Project Explorer
2. Toolbar
3. Editor
4. Console

6.2.1 The Project Explorer

In the Project Explorer you will find three sample projects which you can directly use and adapt. In the Project Explorer you can manage, adapt and test all your projects. After a new installation each project contains the following files:

- control_io.c
- control_io.h
- control.h
- main.c
- startup.c
- *.ld

- *.mak
- settings.mak

6.2.2 Toolbar

The Toolbar includes all the tools needed for working with **Control III**.

6.2.2.1 Compiler



- **clean:**
Used to clean the project folder. Clean deletes all the files and folders created by compiling.
- **debug:**
Compiles the currently selected project without optimization. The resulting control.bin is used for debugging. (See Sec. <The Debugger>).
- **release:**
Compiles the currently selected project. The resulting control.bin is optimized for time in order to achieve the fastest possible cycle times for your program code.

6.2.2.2 Debugging



This button is used among other things for starting the debugger. To debug you must set the corresponding port. More information can be found in Sec. <Start the Debugger>.



Information!

If the Control III program is at a 'breakpoint', the entire operating system freezes and the fieldbus interface is no longer processed

6.2.2.3 Configuration tools

The Configuration tools button is used for communicating with the gateway and has the following functions:



Unlock Control:

Tool for unlocking Control III in the gateway (See Sec. <Enabling Control III>).

Download Control:

The file "control.bin" is written to the gateway.

Start Control:

The **Control III** program in the gateway is started.



Information!

To start a new program, after downloading the running program must first be stopped.

Download + Start Control:

The **Control III** program is first stopped, then the new program is loaded into the gateway and then started.

Stop Control:

The **Control III** program is stopped

Set Auto Start:

The Auto Start flag is set. The **Control III** program starts up automatically after each power-on.

Clear Auto Start:

The Auto Start flag is cleared.

Read Flags:

The Control III flags are read and displayed in the console.

Read Flags Cyclically:

The Control III flags are read and the display in the console refreshed cyclically. This allows for example variables to be monitored during run time.

Cycle time:

The current cycle times are displayed.

Reset Cycle Time:

The cycle time of the **Control III** is reset and recalculated.

6.2.3 Editor

In this window the entire C-code is written and adapted. The Editor also supports errors in the 'C' syntax.

6.2.4 Console

The console serves as an information window. It outputs for example error messages or status messages and displays how much memory is being used by the **Control III** program after compiling.

6.3 File information

As already described in <The Project Explorer>, each project folder contains various files. In this section the individual files are described in greater detail.

control_io.c

This file is used for example to break down a C program into multiple sub-modules so that it is clearer and easier to read.

Both functions 'read_bit' and 'write_bit' read and write a corresponding output or input bit.

```
int read_bit      (AASiProcessData idi, int slave_addr, int bit)
                  idi = Slave input data
                  slave_addr = Address of the corresponding slave
                  bit = Slave input bit (0-3)
```

```
void write_bit   (AASiProcessData odi, int slave_addr, int bit, int value)
                  odi = Slave output data
                  slave_addr = Address of the corresponding slave
                  bit = Slave output bit (0-3)
                  value = Output bit value (0 or 1)
```

control_io.h

Header for control_io.c. This file contains the function definitions for 'read_bit' and 'write_bit'.

control.h

The header file 'control.h' contains all data types and library functions. In addition it explains the way the function and are used (See Sec. <Programming Control III>).

main.c

The 'main' function is the "starting point" of the actual program code. It also contains the main loop of the program (for ; ;).

startup.c

This file is used for initializing various memory ranges. The file has no meaning for the user.

****.ld***

This is the linker file and specifies the corresponding memory ranges in the gateway. The file has no meaning for the user.

****.mak***

This file specifies all the information needed by the compiler. The file has no meaning for the user.

settings.mak

In this file the communication port for the gateway is specified. More detailed information can be found in Sec. <Setting the port>.

6.4 Setting the port

Each project folder contains a file called settings.mak `PORT=COM3`. In this file you can set the port for communicating with the gateway. To do this, select the corresponding file and enter the connection to the gateway. If for example you are using COM Port 3 on your PC, then enter `PORT=COM3` in line 30 of the settings.mak. If you are using an Ethernet port, then enter for example `PORT=UDP:192.168.42.149`.

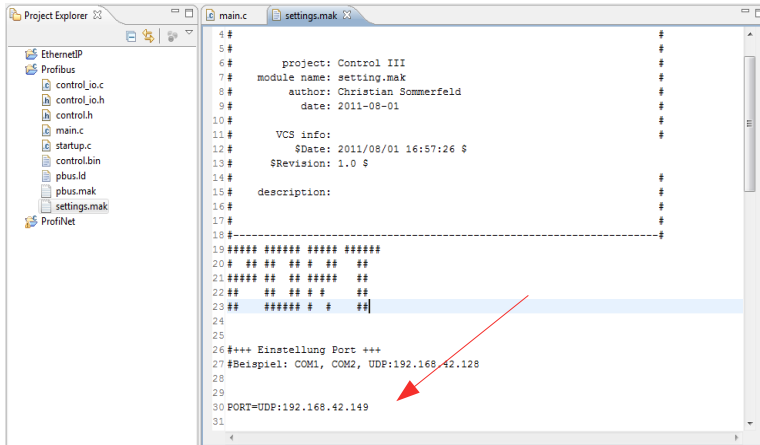


Fig. 6-2. Eclipse settings.mak

6.5 Creating a new project

After a new installation the Project Explorer contains sample projects for each gateway having **Control III**. This makes it possible to start programming directly after installing Eclipse **Control III**.

If you still want to create a new project, proceed as follows:

- Select a new 'C Project' under 'File' -> 'New'.
- Assign a new project name and select an empty 'Makefile Project'.
- Confirm the prompt with 'Finish'.

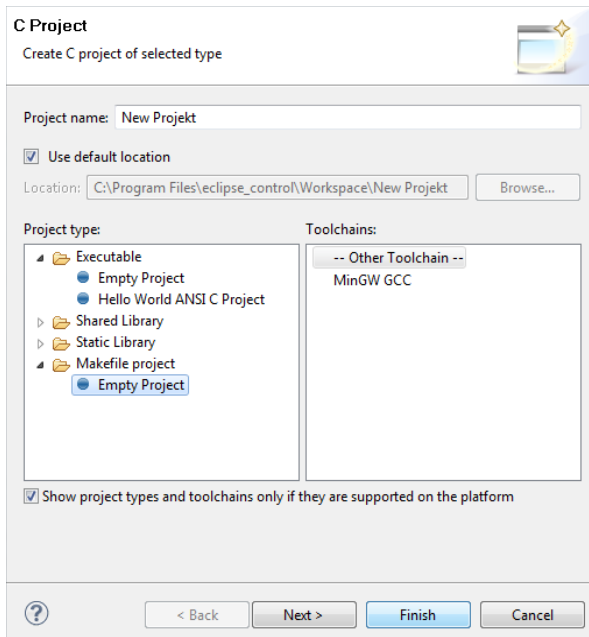


Fig. 6-3. New Control III project

Now you will find a new empty project folder in the Project Explorer. Right-clicking on the new project and 'Import...' allows you to add all the necessary project files for your gateway.

- Select 'File System' and then click on 'Next'.
- Now select 'Browse...' and navigate to your Eclipse installation .../eclipse_control/Templates/
- Now select the gateway you are using.
 - EthernetIP
 - Profibus
 - ProfiNET
- Select 'Select All' and Finish.
- To apply all settings, select 'Yes to All' in the following prompt.

6.6 A sample project

In the following section the complete procedure for writing the code up to starting in the gateway is explained.

6.6.1 The C-code

In the sample program the outputs on a 4I/4O slave having Address 1 are set and cleared one after the other every second. To do this change the main.c as follows:

```
/*-----*
 * include files *
 *-----*/
#include "control.h"
#include "string.h"
#include "control_io.h"

/*-----*
 * local definitions *
 *-----*/

/*-----*
 * external declarations *
 *-----*/

/*-----*
 * public data *
 *-----*/

/*-----*
 * private data *
 *-----*/

static unsigned short system_ticks;
static unsigned short end_timer;

/*-----*
 * private functions *
 *-----*/

static void timer_function ( void )
{
    /* timer interrupt every 10 ms */
    system_ticks++;
}

/*-----*
 * public functions *
 *-----*/

int main ( void )
{
    //initialization of the Debugger
    //ctrl_func.CCtrlBreakpoint();
    unsigned charctrl_flags;
    int i = 0;
    int x = 1;

    AASiProcessData odi[2];
    AASiProcessData idi[2];
    AASiCtrlAccODI acc_odi;
    AASiEcFlags ecflags;
```



```
/* We want to access all odis */
for (i=0;i<32;i++)
{
    acc_odi[i] = 0xFF;
}
cctrl_func.AASiWriteCtrlAccODI ( 0, acc_odi, 0, 64 );

/* init timer function with 10ms ticks */
cctrl_func.CCtrlInitTimer ( 10, timer_function );

/* init watchdog */
//cctrl_func.CCtrlInitWdg( 10 );

    // clear outputs from slave 1
    odi[0][0] = 0x00;
for(;;)
{
    /* trigger watchdog */
    //cctrl_func.CCtrlTriggerWdg();

    /* Define data exchange for AS-i Circuit 1 and 2*/
    cctrl_func.AASiDataExchange(0, odi[0], idi[0], &ecflags);
    cctrl_func.AASiDataExchange(1, odi[1], idi[1], &ecflags);

    //Timer 1 100 * 10ms = 1sec.
    if ( ((unsigned short)(system_ticks - end_timer)) > 100)
    {
        // set and clear outputs circuit=1, slave=1, output=0-3
        if (x == 1) write_bit(odi[0], 1, 0, 1);
        else if (x == 2) write_bit(odi[0], 1, 1, 1);
        else if (x == 3) write_bit(odi[0], 1, 2, 1);
        else if (x == 4) write_bit(odi[0], 1, 3, 1);
        else if (x == 5) write_bit(odi[0], 1, 0, 0);
        else if (x == 6) write_bit(odi[0], 1, 1, 0);
        else if (x == 7) write_bit(odi[0], 1, 2, 0);
        else if (x == 8) write_bit(odi[0], 1, 3, 0);
        x++;

        if (x == 9) x = 1;

        end_timer = system_ticks;
    }

    /* to check Cycletime */
    cctrl_func.CCtrlEvalCycletime();

    /*read flags if we should stop control*/
    cctrl_func.CCtrlReadFlags( &ctrl_flags );
    if ( !( ctrl_flags & CTRL_FLAG_RUN ) )
    {
        return 1;
    }
}
}
```

6.6.2 Compiling

The C-code you have created must now be translated for the processor. To do this, select "release" under the option "Compiler" in the toolbar. A "release" folder is created and your project folder now contains the corresponding binary file 'control.bin'.

6.6.3 Downloading

To load the newly created program into the gateway, go to your project folder and select the file settings.mak and enter the corresponding port for your gateway. For additional information see Sec. <Setting the port>.

Next go to the toolbar and under the configuration tools select: Download Control. If the download was successful the following message appears in the Eclipse console:

```
-----  
++++ CONTROL III ++++  
-----  
  
communication port set to UDP:192.168.42.157.  
  
writing C-Control control.bin to Master ...  
.....  
o.k.  
  
closing control.bin ...  
  
have a nice day.
```

6.6.4 Starting Control III

To start and test the program, go to Toolbar / Configuration tools and select the Start Control button. The following message is displayed:



6.7 The Debugger

A Debugger is a programming tools used for diagnostics and identifying programming errors. If the debugger function is used, the entire operating system is suspended during diagnostics.



Information!

The debug function serves only to test how your program runs. The Debugger cannot be used for diagnosing your hardware configuration.

6.7.1 Initializing

To be able to use the Debugger, it must be initialized in C-code. This is done using the following code line:

```
//initialization of the Debugger  
cctrl_func.CCtrlBreakpoint();
```

Control III Programming in C (small PLC)

Using Eclipse

Using this line stops the operating system and allows Eclipse to communicate with the processor. At this point the Debugger is given the program data and it jumps to the next user-defined breakpoint of the **Control III** program.



Information!

The Debugger should not be initialized in the main loop of the program code (;), since this breakpoint is not controlled by Eclipse and therefore cannot be stopped.

6.7.2 Debugger overview

The screenshot displays the Eclipse IDE with the following components:

- 1. Control Panel:** Located at the top left, it shows the status of the debugger, including the current thread and the state of the program (e.g., "Thread [1] (Suspended Breakpoint hit)").
- 2. Tasks:** Located in the middle left, it shows the list of tasks and the current task being executed.
- 3. Debugger Overview:** Located at the top right, it shows the variables and registers of the program, including the current value of each variable.
- 4. Code Overview:** Located in the middle right, it shows the source code of the program with a breakpoint set at line 101.
- 5. Disassembly:** Located at the bottom right, it shows the assembly instructions corresponding to the source code.
- 6. Console:** Located at the bottom left, it shows the output of the program, including the remote hardware breakpoint limit and the target remote address.

1. Control Panel
2. Tasks
3. Debugger Overview
4. Code Overview
5. Disassembly
6. Console

6.7.2.1 The Control Panel



The Control Panel is used for controlling the Debugger.

Resume (F8):

The 'Resume' button causes the **Control III** program to continue until it reaches a new breakpoint. If no new breakpoint is reached within 10 seconds, the Debugger in Eclipse automatically stops and a corresponding message appears in the console.



Information!

When using the Debugger no delays of longer than 10 seconds are possible, since the Debugger is automatically stopped. This is for quitting the Debugger when communication is absent or incorrect.

Terminate (Ctrl + F2):

Terminate quits the Debugger and allows the **Control III** program to continue running even if there are no breakpoints in the code.

Step Into (F5):

The 'Step Into' button is used for jumping by one program line in the code.



Information!

The StepInto function can only work reliably if the program code has been translated as "debug".

Step Over (F6):

'Step Over' is used for example to skip a function call.

6.7.2.2 Tasks

In this window all programs being used which are controlled by Eclipse are displayed. This window has no great meaning for the user.

6.7.2.3 Debugger Overview

| Name | Value |
|-----------------|------------|
| (0)- ctrl_flags | 0 |
| (0)- i | 0 |
| (0)- x | 1 |
| odi | 0x2000ef54 |
| idi | 0x2000ef14 |
| acc_odi | 0x2000ef94 |
| (0)- ecflags | 48879 |

Variables:

In the Variables tab of the “Debugger overview” all the existing variables and their values can be displayed. Right-clicking in the window allows you to use ‘Add Global Variables’ to add additional variables to the view. The value of the variables is always displayed in hex format. The view can be changed to binary or decimal by right-clicking on ‘Format’.

Breakpoints:

The Breakpoints tab contains an overview of all breakpoints controlled by and used in Eclipse including the program line. Individual breakpoints can be deleted by right-clicking.

6.7.2.4 Code overview

This window allows you to always see at which point in the program code the Debugger is currently located. Double-clicking next to the corresponding code line allows you to set or delete a breakpoint.



Information!

As soon as the Debugger is started you first see an empty window with the text “Source not found.” At this point the Debugger is in the operating system and does not know the associated C-code.

6.7.2.5 Disassembly

Like the Code overview window, the Disassembly window shows exactly where the Debugger is currently. Here both the memory address and the associated assembler code are displayed.

6.7.2.6 Console

The console serves as an output window and tells you about the Debugger status.

6.7.3 Start the Debugger

The Debugger is started from the Control Panel.



Before you can establish a connection with the target, you must define the port for the gateway under ‘Debug-Configuration’. To do this, click on the ‘Commands’ tab and under target remote enter the port. (e.g. target remote UDP:192.168.42.33 or COM3).

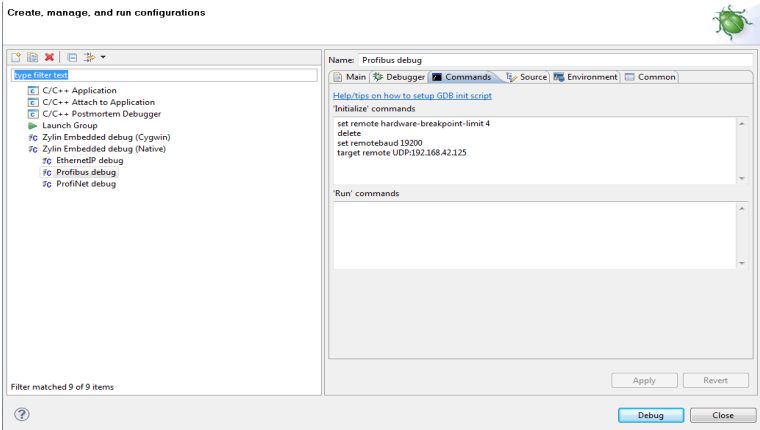


Fig. 6-4. Debug Configuration...

6.7.4 Example

To illustrate the Debugger, here is a sample program which sets an output bit (0-3) of the slave having Address 1 in every main loop pass. Following is the corresponding program code.

```
-----*
 * include files *
-----*/
#include "control.h"
#include "string.h"
#include "control_io.h"

/*-----*
 * local definitions *
-----*/

/*-----*
 * external declarations *
-----*/

/*-----*
 * public data *
-----*/

/*-----*
 * private data *
-----*/

static unsigned short system_ticks;
static unsigned short end_timer;

/*-----*
 * private functions *
-----*/

static void timer_function ( void )
{
    /* timer interrupt every 10 ms */
    system_ticks++;
}

/*-----*
 * public functions *
-----*/
```

```
int main ( void )
{
    //initialization of the Debugger
    cctrl_func.CCtrlBreakpoint();

    unsigned charctrl_flags;
    int      i = 0;
    int      x = 1;

    AASiProcessData odi[2];
    AASiProcessData idi[2];
    AASiCtrlAccODI acc_odi;
    AASiEcFlags ecflags;

    /* We want to access all odis */
    for (i=0;i<32;i++)
    {
        acc_odi[i] = 0xFF;
    }
    cctrl_func.AASiWriteCtrlAccODI ( 0, acc_odi, 0, 64 );

    /* init timer function with 10ms ticks */
    cctrl_func.CCtrlInitTimer ( 10, timer_function );

    /* init watchdog */
    //cctrl_func.CCtrlInitWdg( 10 );

    // clear outputs from slave 1
    odi[0][0] = 0x00;
```



```
for(;;)
{
    /* trigger watchdog */
    //cctrl_func.CCtrlTriggerWdg();

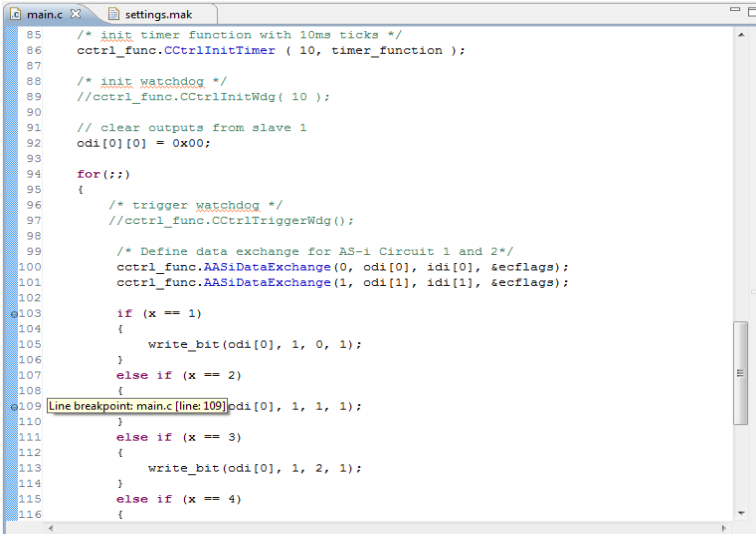
    /* Define data exchange for AS-i Circuit 1 and 2*/
    cctrl_func.AASiDataExchange(0, odi[0], idi[0], &ecflags);
    cctrl_func.AASiDataExchange(1, odi[1], idi[1], &ecflags);

    if (x == 1)
    {
        write_bit(odi[0], 1, 0, 1);
    }
    else if (x == 2)
    {
        write_bit(odi[0], 1, 1, 1);
    }
    else if (x == 3)
    {
        write_bit(odi[0], 1, 2, 1);
    }
    else if (x == 4)
    {
        write_bit(odi[0], 1, 3, 1);
        x = 1;
    }
    x++;

    /* check Cycletime */
    cctrl_func.CCtrlEvalCycletime();

    /*read flags if we should stop control*/
    cctrl_func.CCtrlReadFlags( &ctrl_flags );
    if ( !( ctrl_flags & CCTRL_FLAG_RUN ) )
    {
        return 1;
    }
}
}
/*-----*
 * eof *
 *-----*/
```

Now create two breakpoints by double-clicking in the margin of the program code on the corresponding code line. In our example we use both lines 103 and 109. The line in which a breakpoint was added is indicated by a period. These lines are shown later in the Debugger overview under Breakpoint and this information conveyed to the processor during initialization.



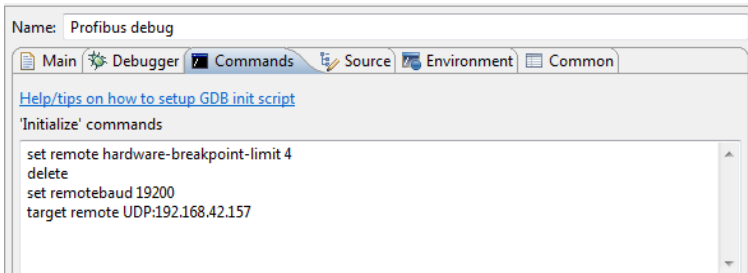
```
85 /* init timer function with 10ms ticks */
86 ctrl_func.CCtrlInitTimer ( 10, timer_function );
87
88 /* init watchdog */
89 //ctrl_func.CCtrlInitWdg ( 10 );
90
91 // clear outputs from slave 1
92 odi[0][0] = 0x00;
93
94 for (;;)
95 {
96     /* trigger watchdog */
97     //ctrl_func.CCtrlTriggerWdg ();
98
99     /* Define data exchange for AS-i Circuit 1 and 2*/
100    ctrl_func.AASiDataExchange(0, odi[0], idi[0], &ecflags);
101    ctrl_func.AASiDataExchange(1, odi[1], idi[1], &ecflags);
102
103    if (x == 1)
104    {
105        write_bit(odi[0], 1, 0, 1);
106    }
107    else if (x == 2)
108    {
109        Line breakpoint: main.c [line:109]odi[0], 1, 1, 1);
110    }
111    else if (x == 3)
112    {
113        write_bit(odi[0], 1, 2, 1);
114    }
115    else if (x == 4)
116    {
```

Fig. 6-5. Marking a breakpoint in the program code

Now compile the program by clicking on 'Debug' in the toolbar under Compiler. A new control.bin file without optimization is created. Load this file into the gateway as already described under <Editor>.

6.7.4.1 Starting the Debugger

Before you can begin debugging, you must set the port. Click in the toolbar under Debugger on 'Debug Configurations...' Here open the 'Commands' tab and under 'target remote' enter your port (e.g. UDP:192.168.42.157 or COM1) for the gateway and click on 'Apply'.



Of your **Control III** program is already at a breakpoint, you can start the Debugger directly from 'Debug'. A breakpoint is displayed in the gateway with number '79'.

79
breakpoint

If this is not the case, quit entry with 'Close' and start your program. You can now directly start the Debugger by clicking for example on 'Profibus debug' in the toolbar.

6.7.4.2 Using the Debugger

After you have started the Debugger, it is configured for the application. The Debugging window is automatically opened by Eclipse. Under Breakpoints you now see the first breakpoint which is run by the program code. This is the Debugger initialization. In this case you are shown an empty window. Click on 'Resume'. The program code is run up to the corresponding line with the first breakpoint and stopped.

```

98
99 /* Define data exchange for AS-1 Circuit 1 and 2*/
100 ctrl_func_AASiDataExchange(0, odi[0], idi[0], secflag);
101 ctrl_func_AASiDataExchange(1, odi[1], idi[1], secflag);
102
103 if (x == 1)
104 {
105     write_bit(odi[0], 1, 0, 1);
106 }
107 else if (x == 2)
108 {
109     write_bit(odi[0], 1, 1, 1);
110 }
111 else if (x == 3)
112 {
113     write_bit(odi[0], 1, 2, 1);
114 }
115 else if (x == 4)
116 {
117     write_bit(odi[0], 1, 3, 1);
118 }
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
```

the corresponding file (control_io.c) and continues 'Debug-mode' from the corresponding location. If you want to jump to the next breakpoint, click again on the 'Resume' button. Quit the Debugger by clicking on 'Terminate'.

7. Technical Data

The following section provides an overview of all the key technical data for Control III.

7.1 Overview

- 28 kB program memory (can be split between ROM and RAM)
- 1kB non-volatile parameters
- 256 bytes of flag memory
- Control program and parameters also on chip card
- A configurable timer interrupt which can be used to create any number of timers
- Programmable timer times from 1 to 2^{32} ms
- Up to 248 I/O and 248 analog values using AS-i slaves
- Unique 32-bit ID in the device
- Simple determination of the cycle time
- Eclipse with GCC and GDB as a complete development environment

7.2 Flags

The flag area is transparent and administered by the user.

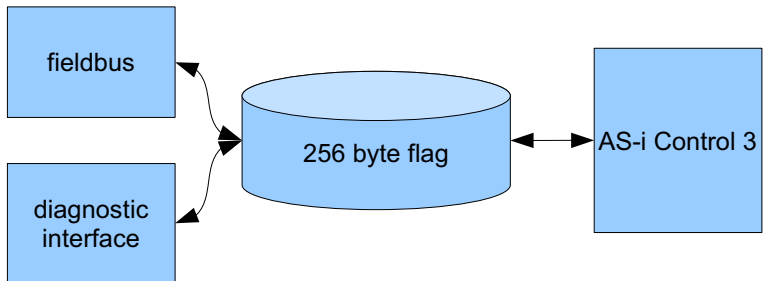


Fig. 7-8. Flag area

7.3 Non-volatile parameters

The non-volatile parameters are transparent and administered by the user.

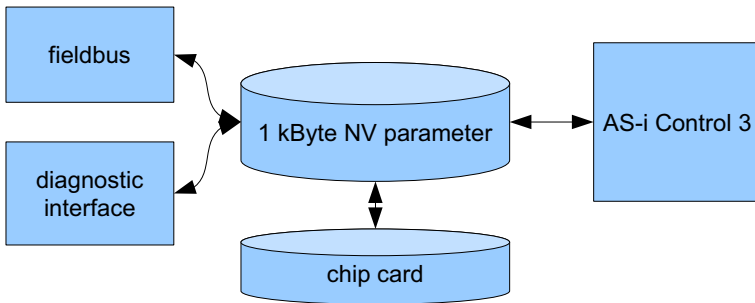


Fig. 7-9. Non-volatile parameters

7.4 Access rights to the output data area

Fieldbus and the Control III program can both set outputs at the same time. Access rights can be assigned by the bit or by channel.

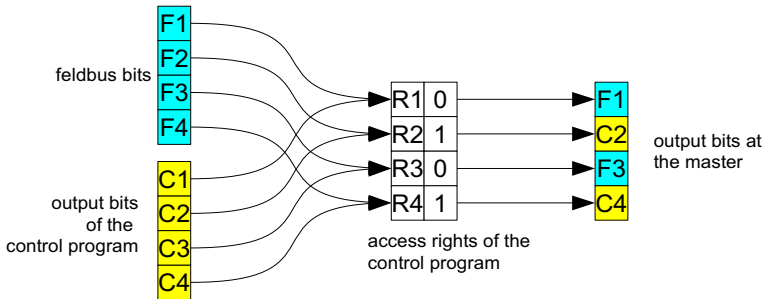


Fig. 7-10. Access rights for the output data area

8. Error Messages

This section is intended to help you to troubleshoot and resolve any problems which may arise.

8.1 error: control not activated!

If the Eclipse console displays the following message, Control III needs to be enabled for your gateway (see Sec. <Enabling Control III>).

```
-----  
++++ CONTROL III ++++  
-----  
  
communication port set to UDP:192.168.42.149.  
  
error: control not activated!  
  
have a nice day.
```

Fig. 8-11. error: control not activated

8.2 error: wrong control version

If the message 'error: wrong control version' is displayed in the Eclipse console, you are using a gateway having a different Control version which is not capable of 'C' programming. In this case refer to manufacturer's support.

```
-----  
++++ CONTROL III ++++  
-----  
  
communication port set to COM4.  
  
error: wrong control version!  
  
have a nice day.
```

Fig. 8-12. error: wrong control version!

8.3 Launching problem

This error message indicates that Eclipse is not associated with your project. Simply click in the Editor window and make your entry again.

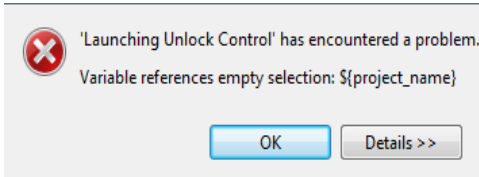


Fig. 8-13. Launching problem

8.4 No or an incorrect cycle time is displayed.

If you read the cycle time on Eclipse and a value of 0 keeps appearing, the following line is missing in the program code:

```
/* check Cycletime */  
cctrl_func.CCtrlEvalCycletime();
```

8.5 The gateway doesn't stop at a breakpoint.

If the gateway doesn't stop at a breakpoint, it is possible that the auto-start flag is set or the following line for initializing the Debugger is missing in the program code:

```
//initialization of the Debugger  
cctrl_func.CCtrlBreakpoint();
```

8.6 The program always goes to the same breakpoint.

If this happens and there is no Debugger initialization in the code, the error is in the program code. The most frequent cause is an un-initialized pointer. Please check your program code. If you have also set the Autostart flag, you can perform a reset when starting the gateway. To do this, press both the 'Mode' and 'Set' keys while turning on the gateway. The existing program code is deleted and the gateway starts back up.

8.7 The Control III can't affect any outputs.

Check whether the slaves are correctly projected and the gateway indicates no configuration error. If needed, project the slaves again. If the error persists, you may not have access rights. These are assigned by the following function:

```
cctrl_func.AASiWriteCtrlAccODI ( ... );
```

Additional information can be found in Sec. <AS-i Read Ctrl Acc AODI>.