

**Instruction manual
Universal Fieldbus-Gateway
UNIGATE[®] IC - DeviceNet**



1630 W. Diehl Rd.
Naperville, Illinois 60563
+1 630 245-1445, +1 630 245-1717 FAX
www.gridconnect.com

1	General introduction	7
2	The UNIGATE® IC	8
2.1	Technical introduction	8
2.2	Availability	8
2.3	Firmware	8
2.4	The serial standard interface	8
2.5	The synchronous interface	8
2.6	The Debug-interface	8
2.7	UNIGATE® IC hardware survey	9
3	Hardware design	10
3.1	Ports	10
3.2	Pinout	10
3.2.1	-Boot enable	11
3.2.2	Load out	11
3.2.3	Data out	11
3.2.4	Data In	11
3.2.5	Load In	11
3.2.6	Clock	11
3.2.7	-Reset In	11
3.2.8	LED-DN	11
3.2.9	-Config Mode	11
3.2.10	DbgTX, DbgRX	11
3.2.11	TE	11
3.2.12	TX, RX	11
3.3	Software	12
3.4	Basic line of proceeding	12
3.5	Connection examples	12
3.6	Layout examples	15
4	The serial interface	17
4.1	Overview	17
4.2	Initialization of the serial interface	17
4.3	Use of the serial interface	17
4.4	Further operation modes	17
5	Synchronous interface	18
5.1	Overview of the synchronous serial interface	18
5.2	Script-example	18
6	The Debug-interface	19
6.1	Overview of the Debug-interface	19
6.2	Starting in the Debug-mode	19
6.3	Communication parameter for the Debug-interface	19

6.4	Possibilities with the Debug-interface	19
6.5	Commands of the Debug-interface	19
7	Script and configuration	20
7.1	Overview	20
7.2	The configuration mode	20
7.3	Update the script	20
8	Generating a script	21
8.1	What is a script?	21
8.2	Memory efficiency of the programs	21
8.3	What can you do with a script device?	21
8.4	Independence of buses	21
8.5	Further settings at the gateway	21
8.6	The use of the Protocol Developer	22
8.7	Accuracies of the baud rates at UNIGATE IC	22
8.8	Script processing times	23
9	DeviceNet.	24
9.1	Setting the DeviceNet-address	24
10	Firmware-update.	26
10.1	Overview	26
10.2	Adjusting the firmware-update-mode	26
10.2.1	Adjustment by hardware.	26
10.2.2	Adjustment by software	26
10.3	Execution of the firmware-update	26
10.4	Note on safety	26
10.5	Operation mode of the IC	26
11	Technical data	27
11.1	Mechanics of the UNIGATE® IC	27
11.1.1	General dimensions of UNIGATE® IC.	27
11.1.2	Dimensions UNIGATE® IC-DeviceNet (old hardware)	27
11.1.3	Dimensions UNIGATE® IC-DeviceNet (new DNL-hardware)	28
11.2	Technical data UNIGATE® IC-DeviceNet	28
12	Accessory	29
12.1	Development board	29
12.2	Adapter RS232	29
12.3	Adapter RS485	29
12.4	FirmwareDownloadTool (FDT)	29
12.5	Protocol Developer	29
12.6	Starterkit IC	29
12.6.1	Quick start	30

13 Appendix - basis board	31
13.1 Overview basis board DeviceNet	31
13.2 Configuration of the UNIGATE® IC	35
13.2.1 DeviceNet	35
13.2.2 RS232/RS485/RS422	35
13.3 Connectors of the basis board	36
13.3.1 Connector to the external device (RS-interface).	36
13.3.2 DeviceNet connector	36
13.3.3 Power supply of the basis board.	36
13.3.4 Shield terminal lead	36
13.3.5 Rotary coding switches	36
13.3.6 Slide switch (RS485/RS232 interface).	37
13.3.7 Slide switch (RS485 termination)	37
13.4 Debug cable for basis board with UNIGATE® IC	37
14 Wiring diagram UNIGATE® IC-basis board DeviceNet	38
15 Servicing	41
15.1 Downloading PC software, EDS files and Script examples etc.	41

1 General introduction

In the past the integration of a fieldbus connection required an enormous effort from the progress engineers. On account of the large variety of communication systems it is not enough to compile the right combination of communication hardware; due to their standards and fundamentals different busses also require the corresponding skills of the engineers.

This does not apply in case of the UNIGATE® IC by Deutschmann Automation any more. All digital functions, software, stack and driver as well as optocoupler are integrated on a UNIGATE® IC in correspondence with the standard. In addition to the reduction of the required size, also different fieldbusses can easily be integrated.

Through the flexible firmware of UNIGATE® IC no software-changes are required on the side of the customer!

Since 1997 Deutschmann Automation has experience in the field of fieldbus gateways; this enormous experience results in the UNIGATE® IC as a consistent sequel of this successful product line.

Terminology

In the entire document and in all parts of the software that is to be used, the terms Input and Output are used. Input and Output are ambiguous, always depending on the viewpoint. We see the fieldbus as central interface and as integral component of your device; therefore in all places it is always referred to data from the viewpoint of the Slave, that is Input data, as data from the Master to the Slave - regardless of the used bus.

Representation of numbers

Numbers in decimal format are always represented without prefix and without suffix as well. Hexadecimal numbers are always marked with the prefix 0x.

2 The UNIGATE® IC

2.1 Technical introduction

The UNIGATE® IC by Deutschmann Automation contains all components that are required for the communication in a fieldbus in one single module. Therefore a developer does not have to take care for that detail any more, only a hardware redesign is necessary in order to integrate the UNIGATE® IC and the required plug connectors.

2.2 Availability

The module is available as DeviceNet. Further fieldbusses are either planned or being worked on. They will only differ in the connections of the busses. The meaning of the general pins 1 - 9 as well as 24 and 26 - 32 will remain unchanged also for further fieldbus implementations. An up-to-date list for all UNIGATE ICs can be found at:
<http://www.deutschmann.de>

2.3 Firmware

UNIGATE® IC is programmed via scripts. On principle any script, that has been developed for a UNIGATE® SC, can also be operated on the UNIGATE® IC.

2.4 The serial standard interface

Intelligent devices, that already feature a micro controller or a microprocessor, are generally supplied with a serial asynchronous interface with a TTL-level. It is directly connected with the TTL-interface of the UNIGATE® IC. For more information on this serial interface see chapter 4 on page 17.

2.5 The synchronous interface

In addition to the standard interface there is also the possibility of the synchronous input and output. That way for instance digital IOs can be connected through shift register components or also analog IOs can be connected through a DA-converter with serial in-/output. For synchronous IOs 256 signals at the most can be used (256 bit). Wiring examples can be found in chapter 3.5 and software examples can be found in chapter 5.1 on page 18. It is also possible to build, for instance digital or analogous I/O-modules, with the customer's device not being equipped with an own controller. The fieldbus IC is also operable autonomously without that controller.

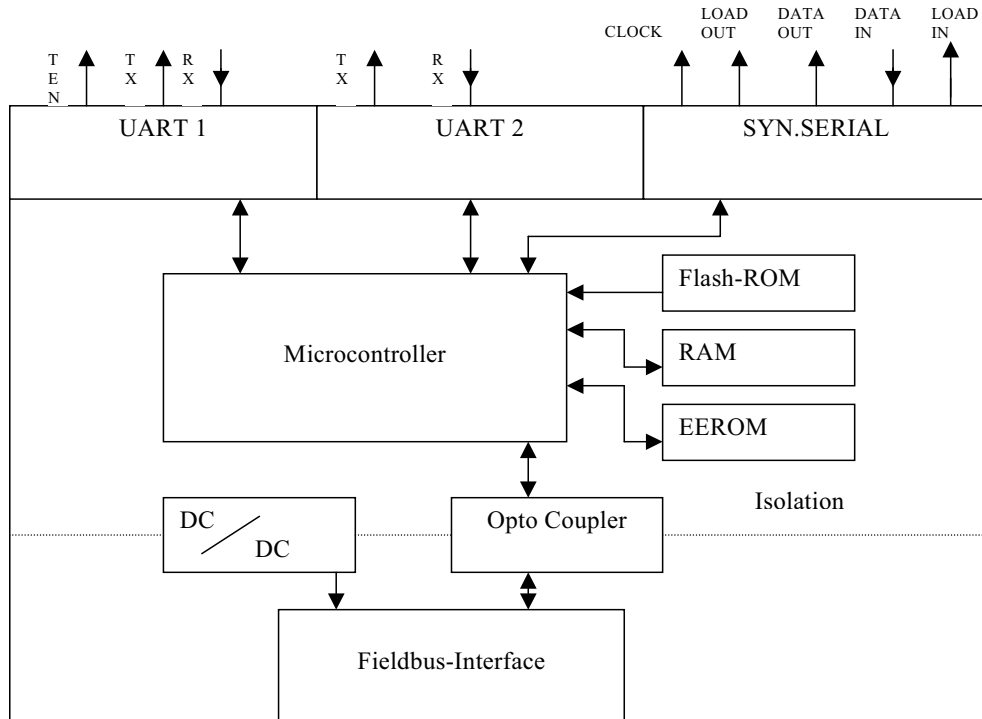
2.6 The Debug-interface

The UNIGATE® IC features a Debug-interface, which allows to process a script step by step and also to monitor or manipulate data. This is indispensable for the development of a script. Usually a script is developed with the software Protocol Developer. For more details take a look at the instruction manual Protocol Developer.

All interfaces can independently be used at the same time.

2.7 UNIGATE® IC hardware survey

The hardware of the UNIGATE® IC consists of some few standard components. The picture below shows the functional structure of the IC.



3 Hardware design

This chapter gives basic advise, that is required in order to load UNIGATE® IC into your own hardware designs. In the following all ports of UNIGATE® IC are described in detail.

3.1 Ports

UNIGATE® IC features 32 pins in its layout as a DIL 32 component. Pin 10 -12 and 21 - 23 as well are not wired due to the electrical isolation. The exact mechanical dimensions can taken from chapter 11 on page 27.



In the layout boreholes for ALL 32 pins have to be planned.

3.2 Pinout

Pin	Technical specifications	Name	Description	Remark
1*	5V ± 5% < 75mA	Vcc	+ 5 V voltage supply	
2	IN _{Logic}	-BE	-boot enable	internally pulled up with 10 kΩ
3	OUT _{Driver}	Load out	strobe signal for synchronous, serial interface	
4	OUT _{Driver}	Data out	output data for synchronous, serial interface	
5	IN _{Logic}	Data in	input data of the synchronous, serial interface	internally pulled up with 10 kΩ
6	OUT _{Logic}	Load in	input data of the synchronous, serial interface; alternatively strobe signal of the output data	
7	OUT _{Driver}	Clock	clock pulse signal for synchronous, serial interface	
8	IN _{Reset}	-Reset in ¹⁾	reset-input of the IC	internally pulled up with 100 kΩ
9*	connected to pin 1	Vcc	+ 5 V voltage supply	
10-12	nc	nc	no pin available	
13	according to norm	CANL	DeviceNet-signal according to standard	galvanically isolated insulation voltage 1000 Vrms
14	according to norm	CANH	DeviceNet-signal according to standard	galvanically isolated insulation voltage 1000 Vrms
15	according to norm	nc	not connected	
16	according to norm	nc	not connected	
17	according to norm	V+	Voltage supply DeviceNet 24V	
18	according to norm	V-	Voltage supply DeviceNet 0V	
19	according to norm		not available, pin internally connected	
20	according to norm	nc	not connected	
21-23	nc	nc	no pin available	
24*	connected to pin 32	GND	ground supply voltage of the IC	
25	OUT	LED-DN	LED DeviceNet	anode of the green LED
26	IN _{Logic}	-Config Mode	signal to start the configuration mode	internally pulled up with 10 kΩ
27	OUT _{Logic}	DbgTX	serial Debug TX	
28	IN _{Logic}	DbgRX	serial Debug RX	internally pulled up with 10 kΩ
29	IN _{Logic}	RX	serial data RX	internally pulled up with 10 kΩ
30	OUT _{Logic}	TX	serial data TX	
31	OUT _{Logic}	TE	transmit enable	
32*	GND	GND	ground supply voltage of the IC	

* The supply voltage is 5 V +/- 5 %, max. 75 mA DC.

The DeviceNet signals are galvanically isolated. The insulation voltage is 1000 Vrms.

	V _{IL}	V _{IH}
IN _{Reset}	< 0.3V / 5mA	> 1.95V / 10µA
IN _{Logic}	< 0.8V / 0.5mA	> 1.95V / 10µA

	V _{OL}	V _{OH}
OUT _{Logic}	< 0.6V / 1mA	> 3.8V / 0.1mA
OUT _{Driver}	< 0.33V / 4mA	> 3.8V / 4mA

3.2.1 -Boot enable

The IC is started in the firmware update mode with the level GND during the power up process. See also chapter 10 on page 26.

3.2.2 Load out

Strobe signal for the synchronous serial interface. With the positive edge at this output data is taken from the connected shift registers to the physical outputs.

3.2.3 Data out

On this line data is output on the synchronous serial interface. The most significant bit of the data is output first.

3.2.4 Data In

Data is read in on the synchronous serial interface via this signal. The most significant bit of the data is expected first.

3.2.5 Load In

This pin is the strobe signal for the input data of the synchronous serial interface.

3.2.6 Clock

This signal is the clock line for the synchronous serial interface. That signal is equally valid for data input and data output.

3.2.7 -Reset In

- A reset generator (Max 809) is on board; with it in the normal case the reset input is not required. In this case the reset input has to be connected with VCC, in order to avoid interferences (see chapter 3.6).
- If the the customer's application has to initiate a reset of the UNIGATE® IC, then the reset input can also be connected with a reset output of the customer's application instead of connecting it with VCC. Here all specifications of the reset signal, mentioned in chapter 3.2 have to be kept.

3.2.8 LED-DN

A green LED can be connected to this wire from hardware revision C (lower board) or Rev. - (DNL-hardware / single-board solution). This LED flashes in the state "Bus ok, not allocated" and shines shines in the state "Allocated".

3.2.9 -Config Mode

If the pin has the level GND, then the IC starts in the configuration mode.

3.2.10 DbgTX, DbgRX

They are transmission line (Tx) and receive line (Rx) as well of the IC's Debug-interface. For the function description of the Debug-interface see chapter 6 on page 19.

3.2.11 TE

The transmit enable signal allows the connection of RS485 drivers to the IC's serial interface. The signal is set to 5V whenever the IC sends via the line TX.

3.2.12 TX, RX

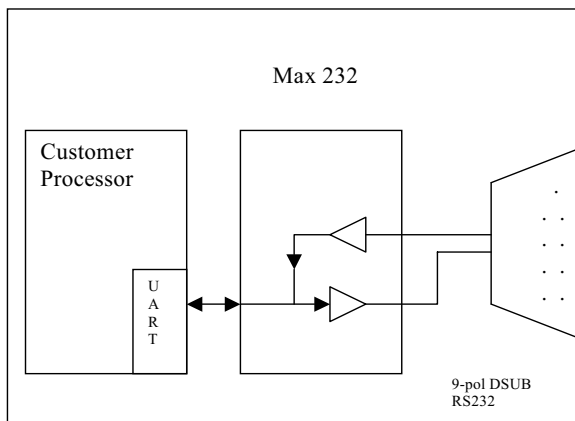
Transmission (Tx) and receive lines (Rx) of the serial interface. This interface is programmable in accordance with the description in chapter 4 on page 17.

3.3 Software

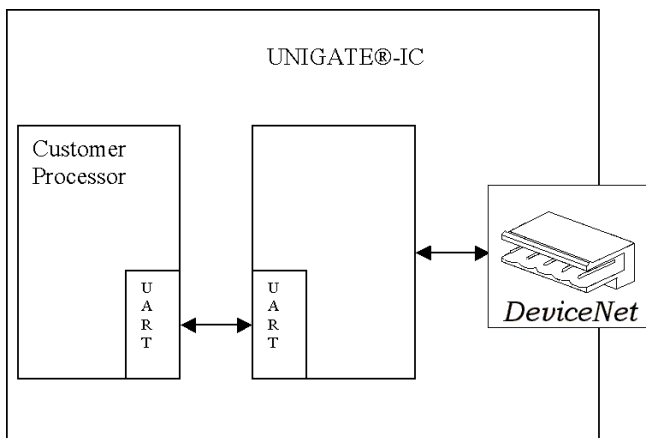
The software executes script-commands, which in turn control the IC's hardware and they process their complete protocol by software. The script itself can be generated by the company Deutschmann Automation or with the software Protocol Developer by yourself. For a detailed description of the script.commands of the Protocol Developer see the instruction manual Protocol Developer and the online documentation concerning script-commands.

3.4 Basic line of proceeding

In theory it is enough to replace the RS232-driver that is included in your application by the UNIGATE® IC.



Your device, which on the whole is supposed to be assembled as shown above, will now be modified in a way that the DeviceNet is available at the 5-pol. socket. However, a hardware redesign is necessary in order to keep the assignment in standard form.



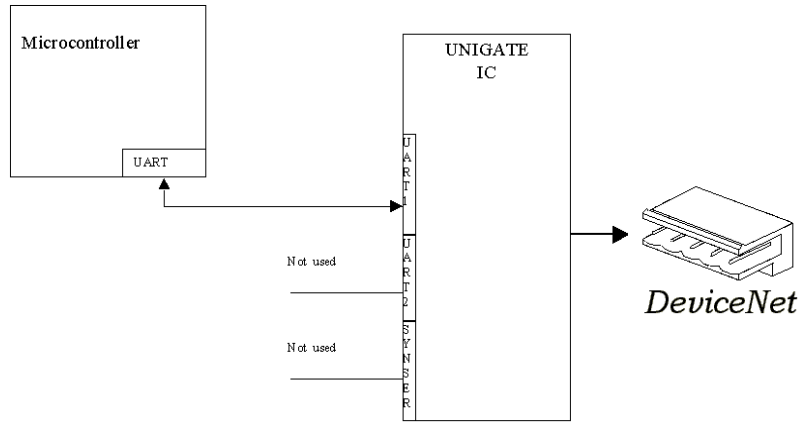
After the RS232-driver has been replaced by the UNIGATE® IC, the DeviceNet is available at the 5-pol. screw-plug connector.

Deutschmann Automation is also offering an appropriate adapter board. With it existing devices can be adapted without re-design; see chapter 12 on page 29.

3.5 Connection examples

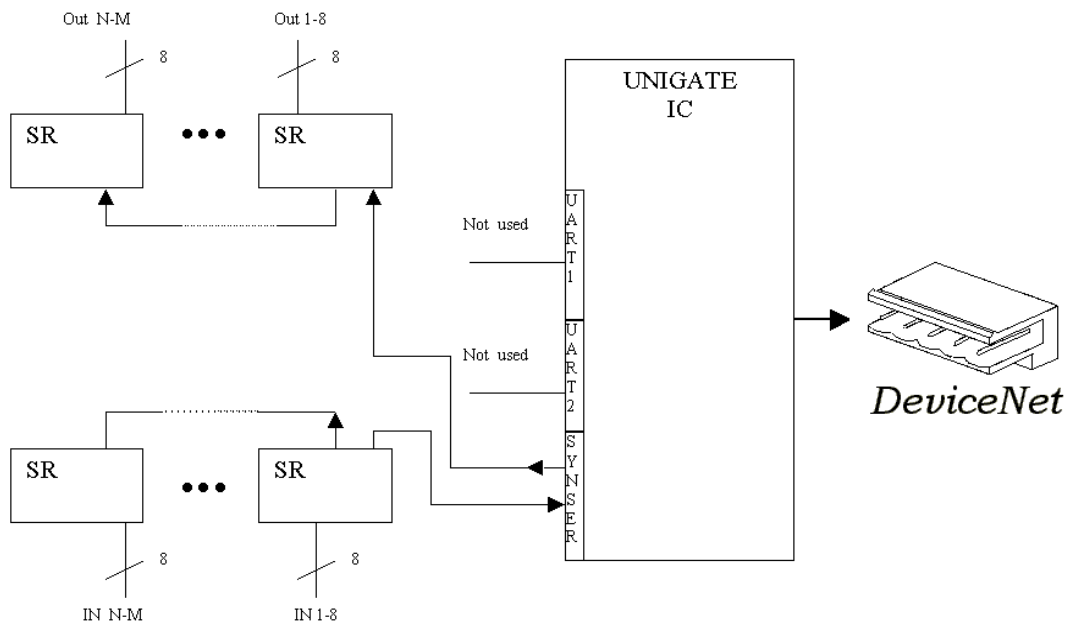
Here you will find some advise that offers help for a re-design. In the following several versions are listed, that should make it easier for you to decide.

Version 1: Use as a pure link module for the bus



The UNIGATE® IC independently processes the communication with the customer's device via the TTL-interface.

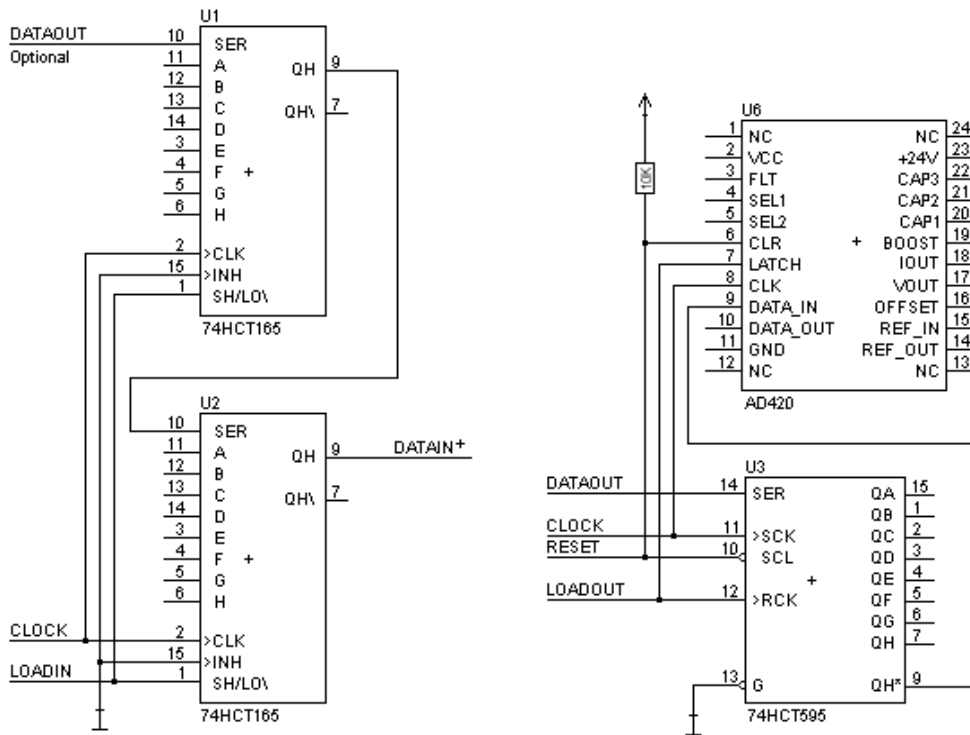
Version 2: Use of UNIGATE® IC for digital or analog I/O-modules



Here only the synchronous serial interface is used, the asynchronous serial interface is basically of no account. If you want to program the script in your completed application, then the use of a connector for the asynchronous interface is recommended. With it you can carry out the ISP-programming.

For this operating mode no additional controller is required on your application!

The following circuit diagram is an example for how shift register components can be connected to the IC.

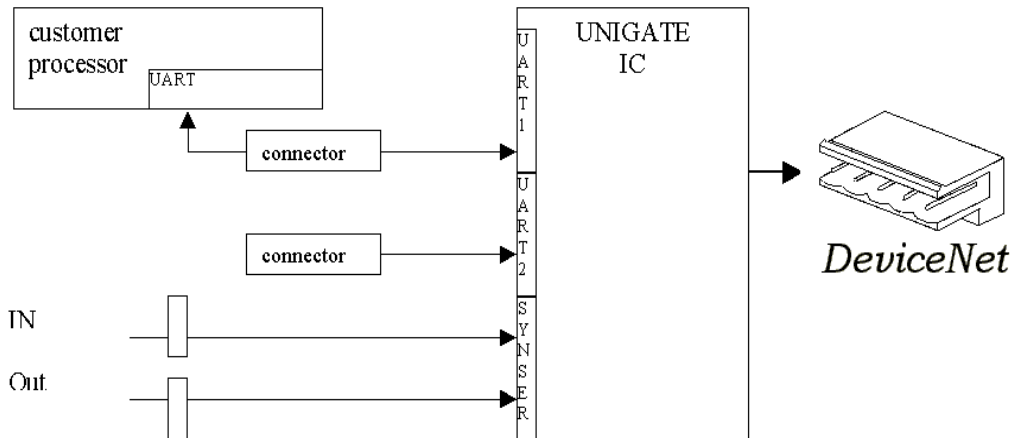


Version 3: Example for digital I/Os

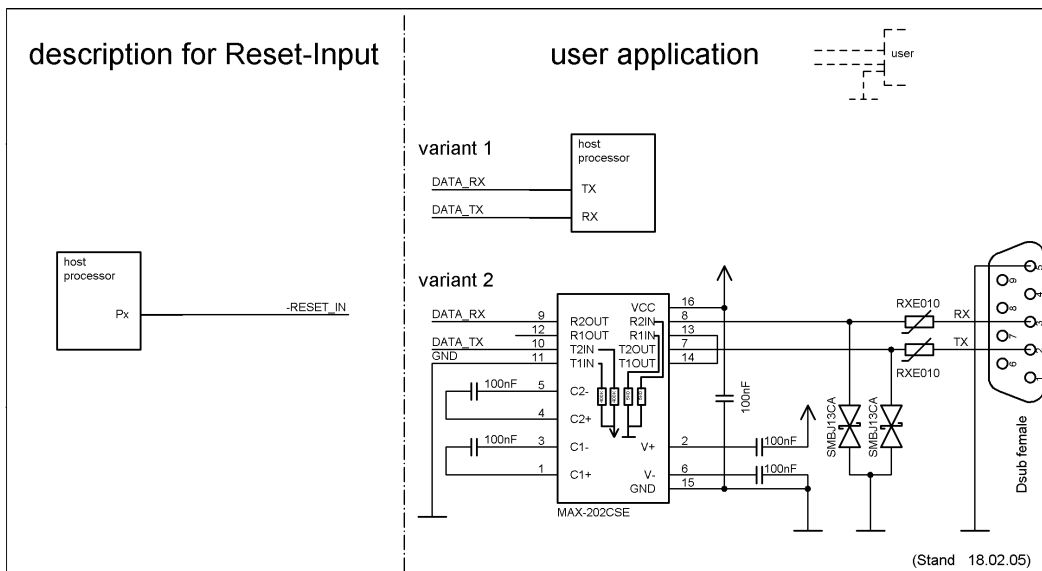
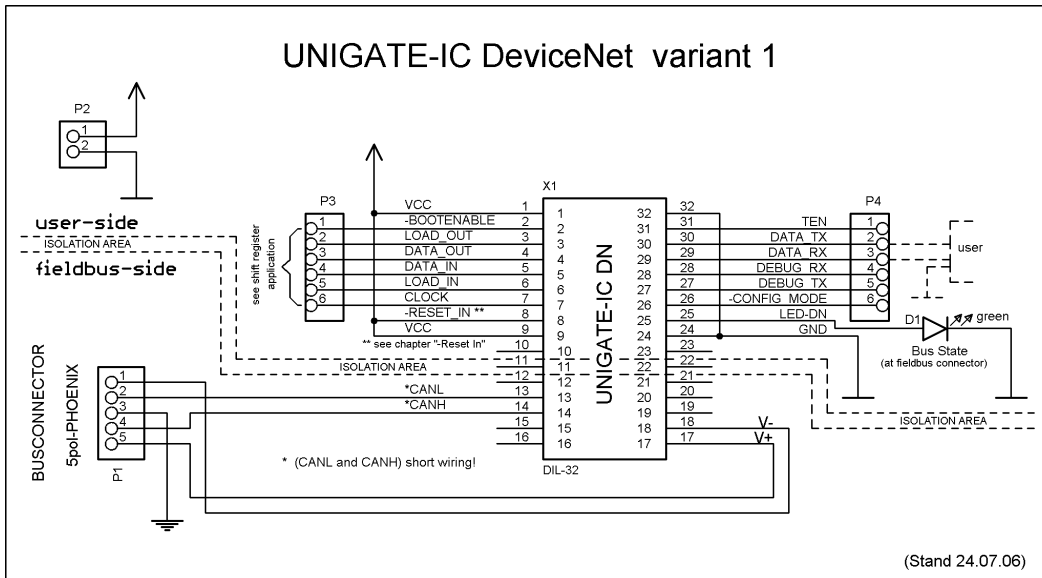
The serial synchronous and the asynchronous interface as well can be operated by UNIGATE® IC at the same time. Here the possibility results that an existing application can be extended by additional digital or analog I/Os.

In chapter 5.2 you find an example for a script, that operates these I/Os.

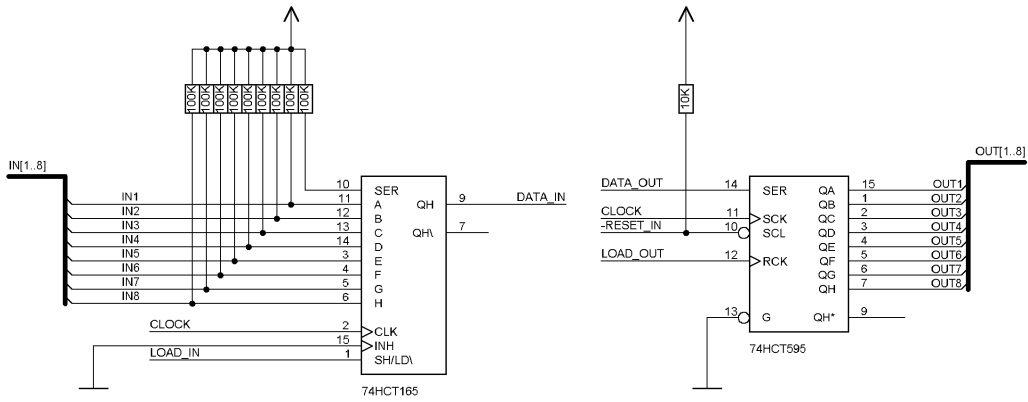
Valid for all versions: A planned plug connection of the serial interface in the application offers the possibility of an update of the firmware or the software via an external connection.



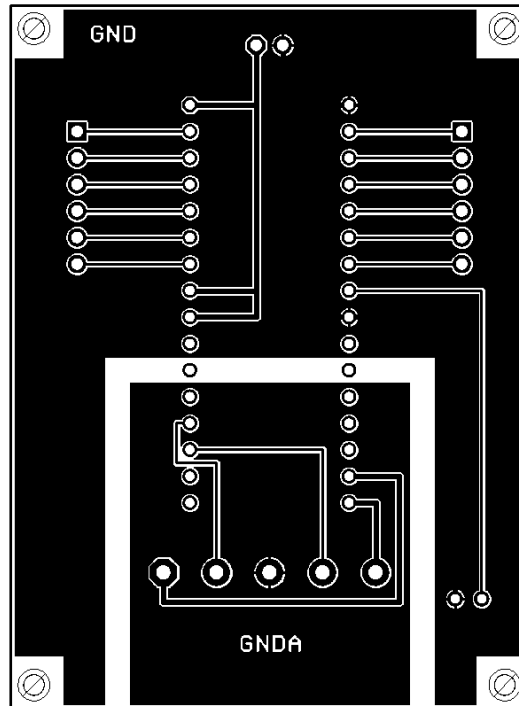
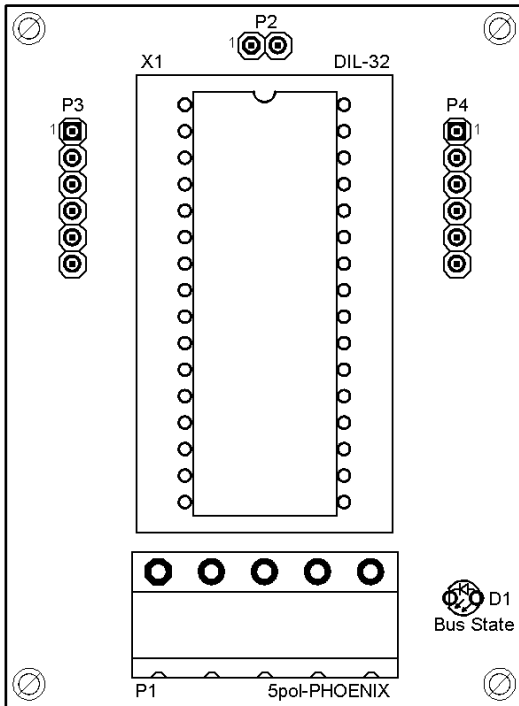
3.6 Layout examples



shift register application



(Stand 26.08.02)



4 The serial interface

4.1 Overview

The serial interface is an important connection between the UNIGATE® IC and the micro controller of your application. The interface is designed in a way so that your application at least does not have to be changed on the software-side. The wide range of services of the UNIGATE® IC's serial interface constitutes the basis for it. The UNIGATE® IC allows to connect controllers with a baudrate of 110 bit to 625 kbit. The baudrate for the communication itself is permanently stored in the module.

Depending on the downloaded script of the UNIGATE® IC, the module carries out actions independently, in order to identify data from the connected device. For customers who already have a software-adaptation at the company Deutschmann Automation, this protocol or script as well can be processed by the IC after an adaptation.

Anyway, the IC will take over the communication with the fieldbus independently.

4.2 Initialization of the serial interface

The initialization of the interface is carried out by script-commands, such as "Set baudrate", "Set databits", "Set parity". For a detailed description of these commands see the online documentation for the Protocol Developer or the instruction manual for the Protocol Developer.

4.3 Use of the serial interface

The serial interface can freely be programmed by the user. Efficient script-commands for sending and receiving data are available; just to mention some possibilities: such as waiting with timeout for a character, waiting for a fixed number of characters or also sending and receiving data in the Modbus RTU. A reference to these commands is offered in the online documentation for the Protocol Developer as well as in the instruction manual for the Protocol Developer.

4.4 Further operation modes

In the modes configuration mode and firmware-update mode the serial interface also serves to configure the standard software or to carry out a firmware-update. More details can be found in chapter 10.5 on page 26.

5 Synchronous interface

5.1 Overview of the synchronous serial interface

The synchronous serial interface is an interface of the IC to clocked shift registers. About this it is not only possible to input or output digital signals but also the addressing of DA- or AD-converters with clocked serial interface is possible as well as the connection of LEDs or reading in rotary switches. Connection examples are stated in chapter 3.

The synchronous serial interface can also be used in products, that do not feature an own micro controller, in order to realize digital IO-modules for instance. Up to 256 signals for input and output each can be processed. The UNIGATE® IC's firmware is responsible for the different amount of input and output signals and takes on control for it.

The data exchange with the script is made with the commands "Set ShiftRegisterInputType / Set ShiftRegisterOutputType" and "Set ShiftRegisterInputBitLength / Set ShiftRegisterOutputBitLength" and "WriteShiftRegister" as well as "ReadShiftRegister".

In order to make the use of the synchronous serial interface as efficient as possible, it is possible to set a „shift register“ and a „bit length“ via the script, whereas both can be changed dynamically within the script by script commands.

As a result the firmware is in the position to control the entire data transfer with the hardware and the data exchange can be carried out as quickly as possible.

At present the shift register types "RiseClk_RiseLoad" and "RiseClk_LowLoad" are implemented, that are required when using for instance the 74595 and 74165 (see also chapter 5.2).

Other types can be complemented very fast and simple - by Deutschmann Automation GmbH.

The shift register type defines the edges or the levels on how data is output to the hardware or how data is read in.

In this case the data exchange is restricted to the script commands "WriteShiftRegister" and "ReadShiftRegister". The clock speed is min. 150kHz, so that for instance a shift register with 32 bit is read in or written in a time period of max. 215µs.

5.2 Script-example

```
var InBuffer: Buffer[2];
Var OutBuffer: Buffer[2];
MoveConst (OutBuffer[0], #0x58#0x21 ),

Set ( ShiftRegisterInputType , RiseClock_FallLoad ) ;
Set ( ShiftRegisterOutputType , RiseClock_RiseLoad ) ;

Set ( ShiftRegisterInputBitLength , 16 ) ;
Set ( ShiftRegisterOutputBitLength , 16 ) ;

WriteShiftRegister ( OutBuffer[0] ) :
ReadShiftRegister ( InBuffer[0] ) ;

// Input data is now in the InBuffer
// 0x58 is applied to the outputs of the analog converter
// 0x21 at the shift register's outputs
```

6 The Debug-interface

6.1 Overview of the Debug-interface

The UNIGATE® IC features a Debug-interface, that allows a step-by-step processing of a script. Normally this interface is only required for the development of a script.

6.2 Starting in the Debug-mode

When applying power to the UNIGATE® IC (power up) the firmware will output the binary character 0 (0x00) after a self-test was carried out on this interface. If the IC receives an acknowledgement via this interface within 500 ms, it is in the Debug-mode. The acknowledgement is the ASCII-character O (0x4F).

With the start in the Debug-mode the further execution of script-commands will be put to a stop.

6.3 Communication parameter for the Debug-interface

The Debug-interface is always operating with 9600 baud, no parity, 8 data bit, 1 stop bit. It is not possible to change this parameter in the Protocol Developer. Please make sure that these settings match those of the PC-COM-interface and that the flow control (protocol) is set to "none" there.

6.4 Possibilities with the Debug-interface

Usually the Protocol Developer is connected to the Debug-interface. With it a step-by-step processing of a script, monitoring jumps and decisions and looking at memory areas is possible. Moreover breakpoints can be set. It basically possesses all characteristics a software-development tool is typically supposed to have. However, it is also possible to carry out a Scrip-update via this interface.

6.5 Commands of the Debug-interface

The commands for the use of the Debug-interface are described in the instruction manual Protocol Developer.

7 Script and configuration

7.1 Overview

In the configuration mode the scripts and configurations, stored in the UNIGATE® IC, can be replaced or updated via the serial interface.

7.2 The configuration mode

If the pin „ConfigMode“ pulled to GND during the PowerUp or Reset, then the UNIGATE® IC starts in the configuration mode. In this mode it is possible to communicate with the IC without processing the regular software. In this mode it is possible to change the UNIGATE® IC's settings of the standard software or to write a new script in the UNIGATE® IC. It shows its start in the configuration mode by issuing a status message, which might look as follows:

```
"IC-DN-SC V4.03[26] (c)dA Script(8k)="Leer" Author="Deutschmann Automation GmbH" Version="1.0" Date=21.08.2001 SN=47110001"
```

7.3 Update the script

The preferred version is the one, where the IC is inserted into the basis board, available from Deutschmann Automation and the Deutschmann tools (such as the software WINGATE with "Write Script" unser "File") are used.

On request the procedures, how to overwrite the included script can be disclosed by Deutschmann Automation in order to automatically replace the script in an application.

8 Generating a script

8.1 What is a script?

A script is a sequence of commands, that are executed in that exact order. Because of the fact that also mechanisms are given that control the program flow in the script it is also possible to assemble more complex processes from these simple commands.

The script is memory-oriented. It means that all variables always refer to one memory area. While developing a script you do not have to take care of the memory management though. The Protocol Developer takes on this responsibility for you.

8.2 Memory efficiency of the programs

A script command can carry out e. g. a complex checksum like a CRC-16 calculation via data. For the coding of this command only 9 byte are required as memory space (for the command itself). This is only possible when these complex commands are contained in a library.

A further advantage of this library is, that the underlying functions have been in practical use for a couple of years and therefore can be described as 'void of errors'. As these commands are also present in the native code for the controller, at this point also the runtime performance of the script is favorable.

8.3 What can you do with a script device?

Our script devices are in the position to process a lot of commands. In this case a command is always a small firmly outlined task. All commands can be put into classes or groups. A group of commands deals with the communication in general. This group's commands enable the gateway to send and receive data on the serial side as well as on the bus-side.

8.4 Independence of buses

Basically the scripts do not depend on the bus, they are supposed to operate on. It means that a script which was developed on a Profibus gateway can also be operated on an Interbus without changes, since the functioning of these buses is very similar. In order to also process this script on an Ethernet gateway, perhaps further adjustments have to be made in the script, so that the script can be executed reasonably.

There are no fixed rules how which scripts have to operate properly. When writing a script you should take into account on which target hardware the script is to be executed, so the necessary settings for the respective buses can be made.

8.5 Further settings at the gateway

Most devices require no further adjustments, except for those made in the script itself. However, there are also exceptions to it. These settings are made by means of the software WINGATE. If you know our UNIGATE-series, you are already familiar with the proceeding with it. An example is the adjustment of the IP-address and the net-mask of an Ethernet-gateway. These values have to be known as fixed values and are not available for the runtime. Another reason for the configuration of the values in WINGATE is the following: After an update of the script these values remain untouched, i. e. the settings that were made once are still available after a change of the script.

Only this way it is also possible that the same script operates on different Ethernet-gateways, that feature different IP-addresses.

8.6 The use of the Protocol Developer

The Protocol Developer is a tool for an easy generation of a script for our script gateways. Its operation is exactly aimed at this use. After starting the program the script that was loaded the last time is loaded again, provided that it is not the first start.

Typical for Windows script commands can be added by means of the mouse or the keyboard. As far as defined and required for the corresponding command, the dialog to the corresponding command is displayed, and after entering the values the right text is automatically added to the script. The insertion of new commands by the Protocol Developer is carried out in a way that existing commands will not be overwritten. Generally a new command is inserted in front of the one where the cursor is positioned. Of course the commands can also be written by means of the keyboard or already written commands can also be modified.

8.7 Accuracies of the baud rates at UNIGATE IC

The baud rate of the serial interface is derived from the processor's crystal frequency. Meanwhile all Script-gateways, except for the MPI-Gateways (20 MHz), are working with a crystal frequency of 40 MHz.

You can enter any desired integer baud rate into the script. After that the firmware adjusts the baud rate, that can be derived the most precisely from the crystal frequency.

The baud rate the gateway is actually working with (BaudIst) can be determined as follows:

$$\begin{aligned} \text{BaudIst} &= (\text{F32} / \text{K}) \\ \text{F32} &= \text{Crystal frequency [Hz]} / 32 \\ \text{K} &= \text{Round}(\text{F32} / \text{BaudSoll}); \\ &\quad \text{Round}() \text{ is a commercial roundoff} \end{aligned}$$

Example:

The actual baud rate is to be calculated, when 9600 baud are pre-set, where the gateway is operated with 40 MHz:

$$\begin{aligned} \text{F32} &= 40000000 / 32 = 1250000 \\ \text{K} &= \text{Round}(1250000 / 9600) = \text{Round}(130.208) = 130 \\ \text{BaudIst} &= 1250000 / 130 = 9615.38 \end{aligned}$$

I. e.: The baud rate actually adjusted by the gateway is 9615.38 baud

The resulting error in per cent can be calculated as follows:

$$\text{Error}[\%] = (\text{abs}(\text{BaudIst} - \text{BaudSoll}) / \text{BaudSoll}) * 100$$

In our example the following error results:

$$\text{Error} = (\text{abs}(9615.38 - 9600) / 9600) * 100 = 0.16\%$$

In practise errors below 2% can be tolerated!

In the following please find a listing of baud rates at a 40 MHz-crystal frequency with the corresponding errors:

4800 baud: 0.16%
9600 baud: 0.16%
19200 baud: 0.16%
38400 baud: 1.35%
57600 baud: 1.35%
62500 baud: 0%
115200 baud: 1.35%
312500 baud: 0%
625000 baud: 0%

8.8 Script processing times

The Script is translated by the Protocol Developer and the consequently generated code is loaded into the Gateway. Now the processor in the Gateway interprets this code. In this case, there are commands that can be processed very fast (e. g. "Set Parameter"). There are also commands, however, that take longer (e. g. copying 1000 bytes). Consequently, for one thing the processing time differs due to the kind of Script command. But the processing time of the Script commands is considerably more determined by the processor time that is available for this process. Since the processor has to carry out several tasks simultaneously (multitasking system) only a part of the processor's capacity is available for the Script processing. The following tasks - in the order of priority - are executed on the processor:

- Sending and receiving data at the Debug-interface (provided that the Protocol Developer has been started on the PC)
- Sending and receiving data at the RS-interface
- Sending and receiving data at the Fieldbus-interface
- Tasks controlled via internal clock (1 ms) (e. g. flashing of an LED)
- Processing of the Script

From experience approximately 0.5 ms can be calculated for each Script line. This value confirmed itself again and again in many projects as a standard value. He is always quite right if the processor has enough time available for the Script processing.

By means of the tasks mentioned above, the following recommendation can be formulated in order to receive a rather fast Script processing:

- Deactivate the Debug-interface (it is the normal case in the serial use)
- Keep the data length at the RS-interface as small as possible. The baud rate is not the problem here, but the amount of characters which are transferred per second.
- Do not unnecessarily extend the data length at the Fieldbus side. Especially at acyclical bus data, if possible do only send them when changes were made. The data length at buses that are configured to a fixed length (e. g. Profibus) should not be longer than absolutely necessary.

If the processing time should be too large in spite of these measures, there is the possibility to generate a customized Script command, that executes several tasks in one Script command. Please contact our support department for this purpose.

9 DeviceNet

At present UNIGATE® IC-DeviceNet supports the data exchange in the mode “polling”. The other possible modes “bit-strobe” and “change of state” are in preparation. In the mode “polling” the UNIGATE® IC is at present restricted to up to 255 bytes input and output. Every combination of input- and output-size is possible. However, the EDS-file will not fit any more. Then it becomes necessary either to work without the EDS-file, which is possible for most systems or the EDS-file that is available from our website has to be modified.

9.1 Setting the DeviceNet-address

There are different possibilities to set the IC's DeviceNet-address.

1. Setting the address through the configuration
The UNIGATE® IC has to be in the configuration mode (see also chapter 7.2 'The configuration mode'). With WINGATE it is now possible to set the address. This address is preserved until it is changed again.
2. Setting the address through the Script
The address can be stored in the Script as well. This proceeding, however, is likely to be of interest for a few applications only, since it is necessary to change the Script in order to also adjust the DeviceNet-address (see also the following Script example).
3. Setting the address through the serial interface
The address can also be transmitted to the IC through the serial interface. Then the address can be set with the Script command "SetByVar". This possibility should be used in case your device has a control front available and the menu of the front can be extended by the setting "DeviceNet-address". The adjustment of the Profibus-address through the serial interface is the most convenient possibility for those applications.
However, as usual for DeviceNet also the baudrate of the DeviceNet should be set through the Dip-switch.
4. Connecting the rotary switches to the shift registers
Rotary switches can be connected to a shift register as well as to our basis board. Now it is possible for the Script to read those switches and to set them as fieldbus-address. Basically the following Script can be used for it.

Script example for the initialization of the Profibus

```
var InSize: word;
var OutSize: word;

Set (FieldbusID, 4) ;
// this parameter can also be set by the command SetByVar
// var DNAddress: long;
// MoveConst( DNAddress, 4) ; or from the shift registers
// SetByVar();

// Define the baudrate the bus is supposed to be operated with.
// here exemplary 125 kBaud
// This has to take place before the bus-start
Set ( BusBaudrate, 125000 );

// Before the bus-start the participant has to be configured.
// Most important is the setting of the data width,
// Here the values are exemplary.
MoveConst ( InSize, 10) ;
MoveConst ( OutSize, 12 );
SetByVar ( BusInputSize, InSize );
SetByVar ( BusOutputSize, OutSize );
// InSize and OutSize are from the IC's point of view!!
// Here the values can be set with Set.

// The ProductCode of the participant can also be determined.
// This has to take place before the bus-start
// ! It is not important to set the command Productcode.
// It is possible to set a fixed ProductCode for a script gateway.
// If this value is set to 0 the gateway calculates its product code by
// 256 * consumed size + produced size
// !!! If you like to set a special ProductCode, you MUST set this command after
// "BusInPutSize" and "BusOutputSize"

BusStart;
// The DeviceNet is ready. From now the Master CAN configure
// the participant.
// However, this does not mean that the Master has already
// opened up a Poll-connection with the Slave.

wait (Bus_Active);
// The Poll-connection has now been set up by the DeviceNet
// Scanner.
// This command might take a very long time and it cannot
// be interrupted!

// Data can be read out from the bus
// As many bytes as available should be read.
var InBuffer: Buffer[100];
Readbus ( InBuffer[0], InSize) ;

// Now it is possible to write data.
// You must not write more bytes than available.
var OutBuffer: Buffer[100];
WriteBus ( OutBuffer[0], OutSize );
```

10 Firmware-update

10.1 Overview

UNIGATE® IC has a 64 kbyte flash memory for the firmware. In the firmware-update-mode the firmware can be replaced via the UNIGATE® IC's serial interface.

10.2 Adjusting the firmware-update-mode

10.2.1 Adjustment by hardware

UNIGATE® IC can be brought to the firmware-update-mode by the hardware. For it the signal -BE (boot enable) has to be pulled to the potential GND during the Power-up-process.

10.2.2 Adjustment by software

If the UNIGATE® IC is in the configuration mode (see chapter 7.2 on page 20) it can be brought to the firmware-update-mode interactively through the command CTRL-F (0x06). After sending the command a security query follows, that has to be answered with J or N (J = Yes, N = No). After a positive confirmation the IC is re-started in the firmware-update-mode.

10.3 Execution of the firmware-update

The safest way for the firmware-update is the use of the basis board combined with the software "FDT.EXE" (firmware-download-tool). These tools are available from Deutschmann Automation (see chapter 12 on page 29).

It is also possible to use the description and the tools of the manufacturer of the controller (TEMIC, 89C51RD2) as well.

10.4 Note on safety

The firmware-update should only be carried out when there is no other possibility left. A firmware-update-process that has already been started CANNOT be undone. With it the previously used firmware is permanently unusable.

10.5 Operation mode of the IC

Standard-operation mode

This mode is required for the regular use of the IC. In this mode the IC will process all script-commands and normally exchange the corresponding user data. The bus as well is operated in this mode through the IC.

Configuration mode

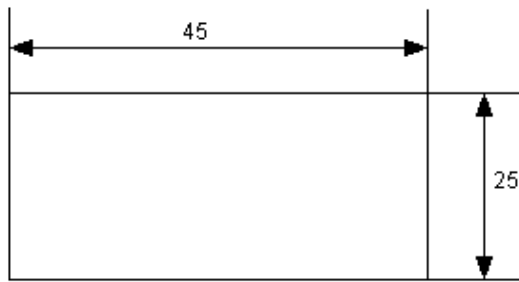
In the configuration mode the UNIGATE® IC will carry out a self-test after the start (or after a reset). After a successful self-test it will wait for further commands. Here it is possible to load a translated script into the device or to initialize the firmware-download-mode.

11 Technical data

In this chapter you will find all necessary technical data on UNIGATE® IC.
All measurements in mm.

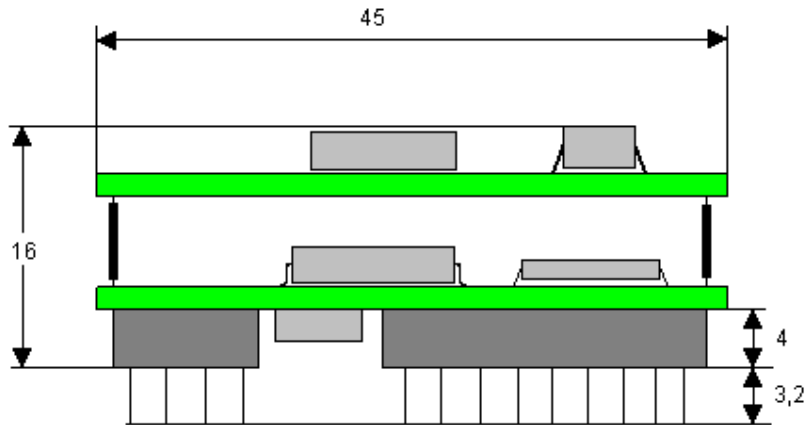
11.1 Mechanics of the UNIGATE® IC

11.1.1 General dimensions of UNIGATE® IC

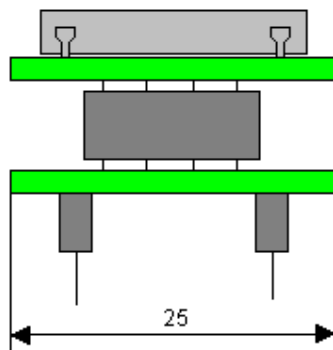


Max. height = 20 mm; including pin

11.1.2 Dimensions UNIGATE® IC-DeviceNet (old hardware)

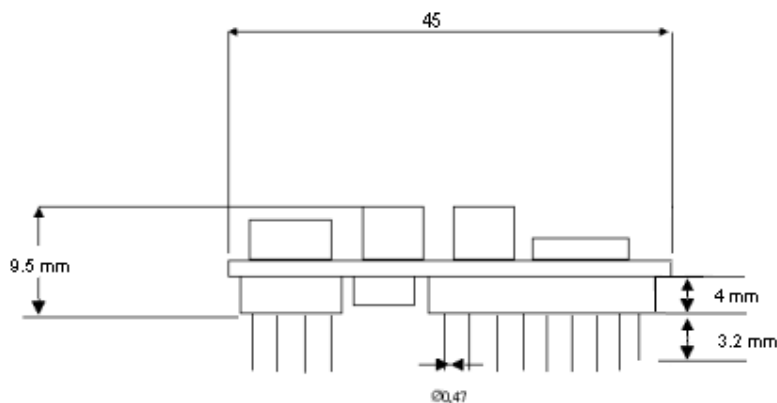


The pins of UNIGATE® IC - DeviceNet are arranged with a grid spacing of 2.54 mm.

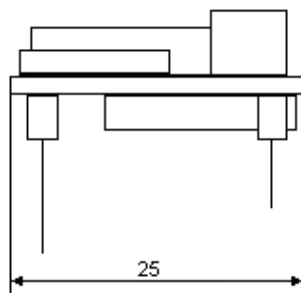


In case you intend to use other fieldbus ICs, the maximum overall height of ≤ 20 mm (including pins) has to be taken into consideration.

11.1.3 Dimensions UNIGATE® IC-DeviceNet (new DNL-hardware)



The pins of UNIGATE® IC are arranged with a grid spacing of 2.54 mm.



In case you intend to use other fieldbus ICs, the maximum overall height of ≤ 20 mm (including pins) has to be taken into consideration.

11.2 Technical data UNIGATE® IC-DeviceNet

Characteristics	Explanation
Supply voltage	5 V \pm 5 %, max. 75 mA DC
Interface	2 UART interfaces, 1 synchronous serial interface
Physical separation -fieldbus-side	Standard
Fieldbus-ID	Adjustable via script
Fieldbus-baud rate	Up to 500 Kbaud (adjustable via script)
Product code	Adjustable via script
I/O-sizes	Adjustable via script at present (0...255 bytes)
UART-baud rate	Up to 625 Kbaud (adjustable via script)
Fieldbus data format	Group 2 slave
Technology	SJA1000
Others	Polling E.g. digital I/Os, analogue signals, shift registers LEDs, switches tec. can be connected externally
Dimensions	45 x 9.5 x 25 mm (WxHxD), old hardware: 45 x 16 x 25 mm (WxHxD)
Installation	32 DIL
Weight	15 G
Operating temperature	-40°C ..+85°C
Storage / transport temperature	-40°C..+125°C
Built-in position	Any

12 Accessory

The following tools are available from Deutschmann Automation.

12.1 Development board

The development board (also called basis board) is a circuit board on which the UNIGATE® IC can be installed. The IC is installed in the ZIF-socket. The ZIF-socket allows an easy insertion and remove of the IC.

Together with the IC this board is almost compatible to 100% with a script gateway of the SC-series. At the regular UNIGATE-SC only the sizes of the I/O-buffers and the available memory are a little smaller.

12.2 Adapter RS232

In an application the adapter RS232 offers the possibility to replace an existing driver MAX 232 (only in DIL-16-housing) by this adapter. This board allows the use of the IC according to chapter 3.4 on page 12. Please note that with it the DeviceNet does not offer a connection conforming to the standards. With a plug adapter, however, at least the operation of DeviceNet is possible.

The hardware is only meant for development purposes. It offers the possibility to make an existing application capable for bus connection in no time and to test the IC's utilizability and functionality.

12.3 Adapter RS485

From the functionality's point of view the RS485 adapter is the same as the RS232 adapter. It offers the possibility to replace a module LS 176 (only in DIL-8-housing) by the IC.

There are the same restrictions as for the RS232 adapter.

12.4 FirmwareDownloadTool (FDT)

The FirmwareDownloadTool is available for download from our homepage: it is required for an update of the firmware. Condition for it is, that a PC can be connected to the serial of the IC. The software describes the procedure of an update itself.

12.5 Protocol Developer

The Protocol Developer is the development environment for scripts, that also contain the Debugger. This software package also contains the documentation to all script-commands. This software is available for download from our homepage at <http://www.deutschmann.de>. The instruction manual for the Protocol Developer, which is available in pdf-format, gives further advise on how to use the software.

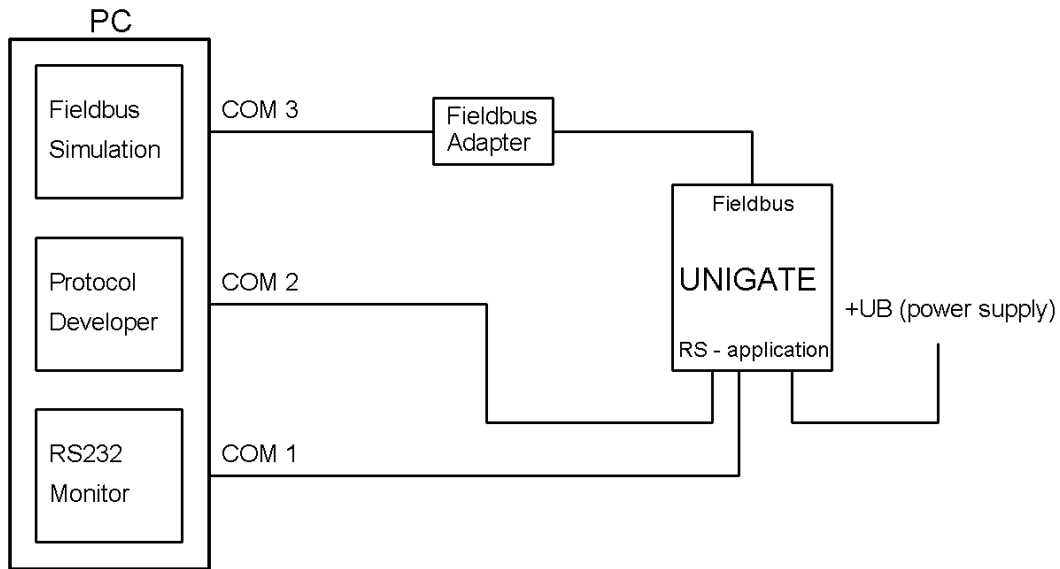
12.6 Starterkit IC

The Starterkit contains

- a UNIGATE® IC for the selected fieldbus
- a UNIGATE® IC-basis board to put on the UNIGATE® IC
- a plug-in power pack to supply the UNIGATE®
- a connection cable
- Software and documentation complete the packet. Additionally a 3-months hotline service is included

The basis board contains a socket to put on the IC, all RS- and fieldbus-sided connectors, switches and LEDs. Actually you cannot tell the difference to a 'normal' UNIGATE® any more. With the basis board your UNIGATE® IC can be connected to your product within a few minutes, in order to prepare the implementation then. Through the basis board you have direct access to all functions of the IC. With the Protocol Developer you write your script quickly and with the IC in the basis board it can be tested effectively. Master simulation of the fieldbus side is available as add-on for the described Starterkit. Besides the corresponding fieldbus-master simulator the required connection cable to the PC and a PC-software for the representation of the RS-data and the fieldbus-data is also supplied.

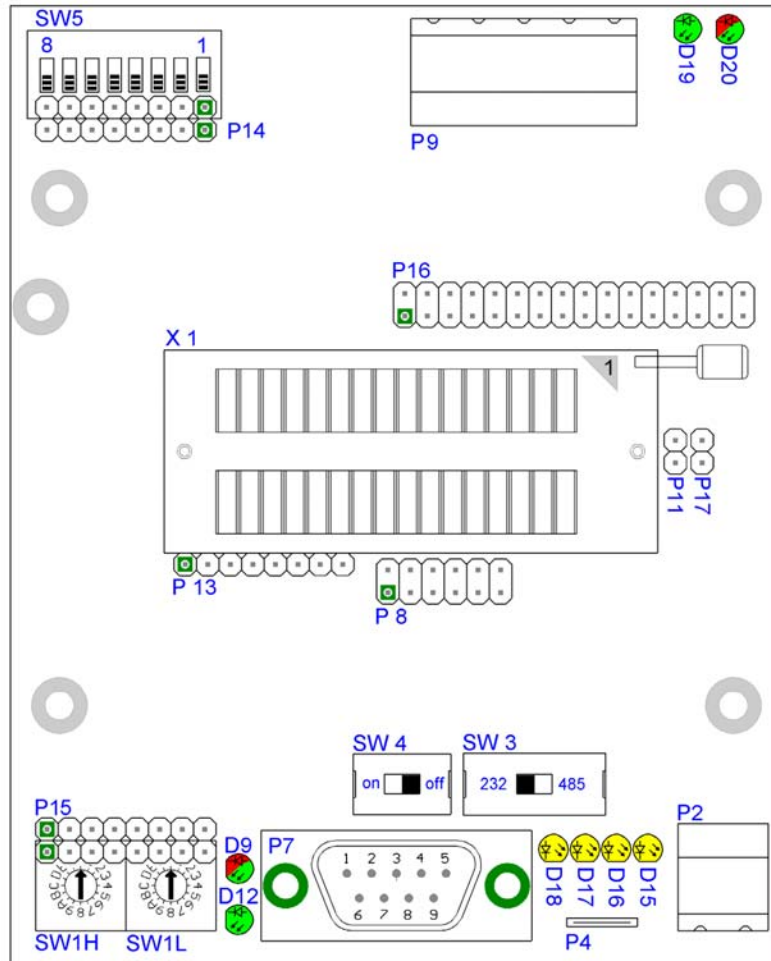
12.6.1 Quick start



For a transparent data exchange you will find example scripts for the respective Fieldbus under "File->New" in the Protocol Developer.

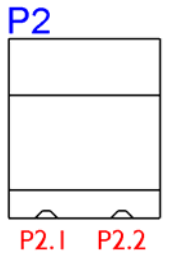
13 Appendix - basis board

13.1 Overview basis board DeviceNet



Slot X 1 (ZIF-socket)

PIN 1 of the IC is located up at the lever of the ZIF-socket.
Never place the IC into the socket back to front!

P 2

Pin	Signal
Pin 1	24 V DC
Pin 2	Ground

The basis board is supplied with voltage through this plug connector.

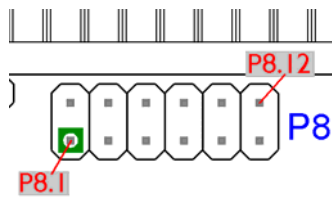
P 4

Earth terminal 6.3 mm for basis board.

P 7

This plug is the basis board's serial connection to the customer's device and the connection to the PC (Debug-interface).

For the pin assignment see chapter 13.3.1 and chapter 13.4.

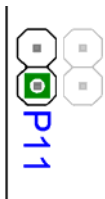
P 8

The illustration shows the arrangement of the pins. On this connector strip the signals of the serial connection between IC and RS-drivers are wired. For an initial development you will probably also use an existing driver in your application. In order to exchange it later on, you can also directly take the signals of the serial interface here.

P 9

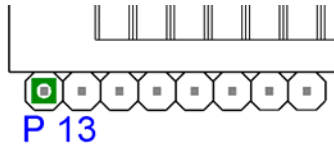
DeviceNet plug connector

For the assignment of the connector see chapter 13.3.2.

P 11

Force Boot. By setting this bridge the Pin BE is dragged to Ground. For the function see chapter 10.2.1.

P 13



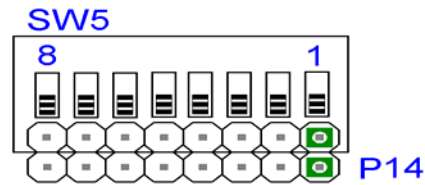
Status signal of the IC
 Plug connector P 13

Pin	Signal
1	Vcc
2	Gnd
3	-RESET
4	RX of the IC (TTL-level)
5	TX of the IC (TTL-level)
6	TE Pin IC (TTL-level)
7	TX Debug of the IC (TTL-level)
8	RX Debug of the IC (TTL-level)

P 14, SW5H, SW5L

Input shift register

For a detailed assignment and for information on which pin is assigned to which bits, see also chapter 5.



Connection	Pin	Meaning
P 14	1	Input 9

	8	Input 16
SW5H	1	Input 25

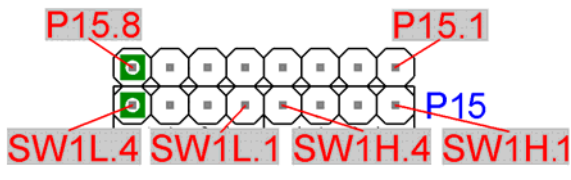
	4	Input 28
SW5L	1	Input 29

	4	Input 32

P 15, SW1H, SW1L

Input shift register

Basically the same applies as for P 14, with the exception that different input bits of the shift registers are wire.



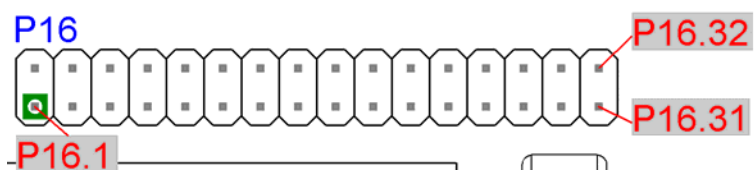
Connection	Pin	Meaning
P 12	1	Input 1

	8	Input 8
SW1H	1	Input 17

	4	Input 20
SW1L	1	Input 21

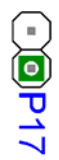
	4	Input 24

P 16



All digital outputs of the shift registers are available here. Additionally the LEDs D9, D15..D18, D20 are connected to the shift registers.

P 17

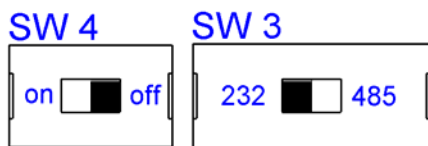


With P17 the UNIGATE® IC can be brought into the configmode. If the jumper is plugged and if the UNIGATE® IC is restarted (by power off and power on or by reset), then the UNIGATE® IC will start in the configmode. In order to use the configmode with Deutschmann software tools the interface of the board has to be in RS232-position and the PC has to be connected with the „normal“ interface, where otherwise your application is connected to. See also chapter 7.2.

SW1H, SW1L, SW5H, SW5L

The rotary switches SW1H, SW1L, SW5H, SW5L are plugged into the base boards and can be removed if required. As a default the rotary switches are plugged in and can be read in through the basis board's shift registers base boards (see also chapter 5 for it).

SW3, SW4



These switches are required for the setting of the serial interface. The switch SW3 is used to switch between interface RS232 and RS485. This is the interface, the customer's device is connected to. The Debug-interface always has RS232-level.

The switch SW4 is of importance only, when it is an RS485-interface. Then this switch can be used to connect the termination of the RS485-bus.

Each switch position can be taken from the illustration.

D12

Power LED

This LED is always supposed to be shining statically green when the board is supplied with voltage.

D9, D15..D18, D20

LEDs that are connected to the shift register components. See also (see also chapter 14 'Wiring diagram UNIGATE® IC-basis board DeviceNet').

D19

LED DeviceNet

The LED flashes in the state "Bus ok, not allocated" and shines in the state "Allocated".

13.2 Configuration of the UNIGATE® IC

UNIGATE® IC is delivered with an empty script.

The configuration of the UNIGATE® IC - DeviceNet is restricted to the setting of the fieldbus address.

13.2.1 DeviceNet

- Vendor-ID: 272
- Device-type: 12 (communication adapter)
- MAC-ID: in accordance script setting or configuration data
- Poll-data length: In accordance with script
- Product code: In accordance with script

13.2.2 RS232/RS485/RS422

- RS-type: RS232
- Start bit: 1
- Data bits: 8
- Stop bit: 1
- Parity: None
- Baud rate: 9600 Baud
- Default setting: This configuration can be changed via the Script.

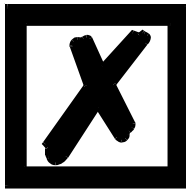
13.3 Connectors of the basis board

13.3.1 Connector to the external device (RS-interface)

The connection cable to the external device must be plugged in at the connector accessible on the underside of the device.

Pin assignment P7 (9-pin Sub-D, plug, debug version)

Pin No.	Name	Function
1		
2	RX/RS485- / RS422- (TX)	Receive signal customer's device
3	TX/RS485+ / RS422+ (TX)	Transmit signal customer's device
4	TX / Diag	Transmit signal Debug interface
5	GND RS	Ground connection, reference for PIN 2+3+6+7
6	RS422- (RX)	
7	RS422+ (RX)	
8	Not connected	Not connected
9	RX / Diag	Receive signal Debug interface



Attention:

In case the RS-interface is NOT potentially divided, "GND" and "supply 0V" are connected internally.

Pin assignment P2 (2-pin screw-/plug-connector)

Pin No.	Name	Function
1	10.8...30 V / DC	10.8...30 V supply voltage
2	0 V / DC	0 V supply voltage

13.3.2 DeviceNet connector

The connector for connection to DeviceNet is located on the upper side of the device.

Pin assignment P9 (5-pin screw-/plug-connector)

Pin No.	Name	Function
1	V-	ØV
2	CAN-L	Dominant Low
3	Shield	Cable shield
4	CAN-H	Dominant High
5	V+	24V power

13.3.3 Power supply of the basis board

The device must be powered with 10.8-30 VDC. The voltage supply is provided via the separate 2-pin screw-plug connector (P2).

Please note that basis boards cannot be operated with AC voltage.

13.3.4 Shield terminal lead

The shield signal for the electronic circuitry is connected to the top-hat rail via the connector provided. The shield signal for the DeviceNet cable shield is not electrically connected to the shield signal of the electronic circuitry for reasons relating to interference immunity.

13.3.5 Rotary coding switches

The rotary coding switches are socketed and can be taken off, in order to alternatively use the pins of the shift register.

13.3.6 Slide switch (RS485/RS232 interface)

This slide switch is used to select whether an RS485 interface or an RS232 interface is connected at the connector to the external device.

13.3.7 Slide switch (RS485 termination)

If the gateway is operated as the first or last physical device in the RS485 bus, there must be a bus termination at this gateway. In order to do this, either a bus terminating resistor in the connector or the resistor (150 Ω) integrated in the gateway must be activated. In order to do this, slide the slide switch to position ON. In all other cases, the slide switch must remain in position OFF. Please refer to the general RS485 literature for further information on the subject of bus terminations.

If the integrated resistor is used, please allow for the fact that this also activates a pull-down resistor (390 Ω) to ground and a pull-up resistor (390 Ω) to VCC.

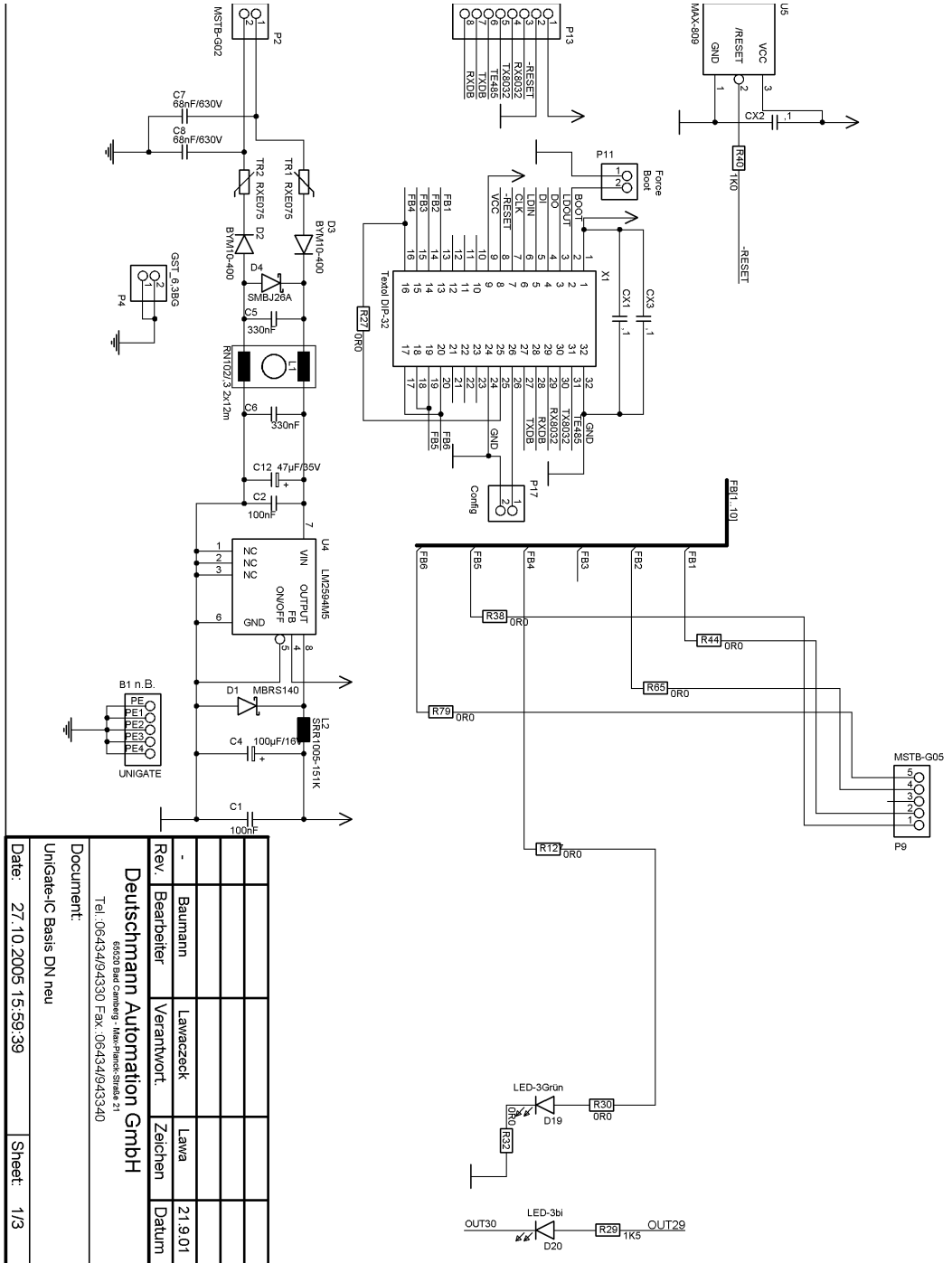
At the RS422-interface the transmission line is terminated. The receive line is always firmly terminated.

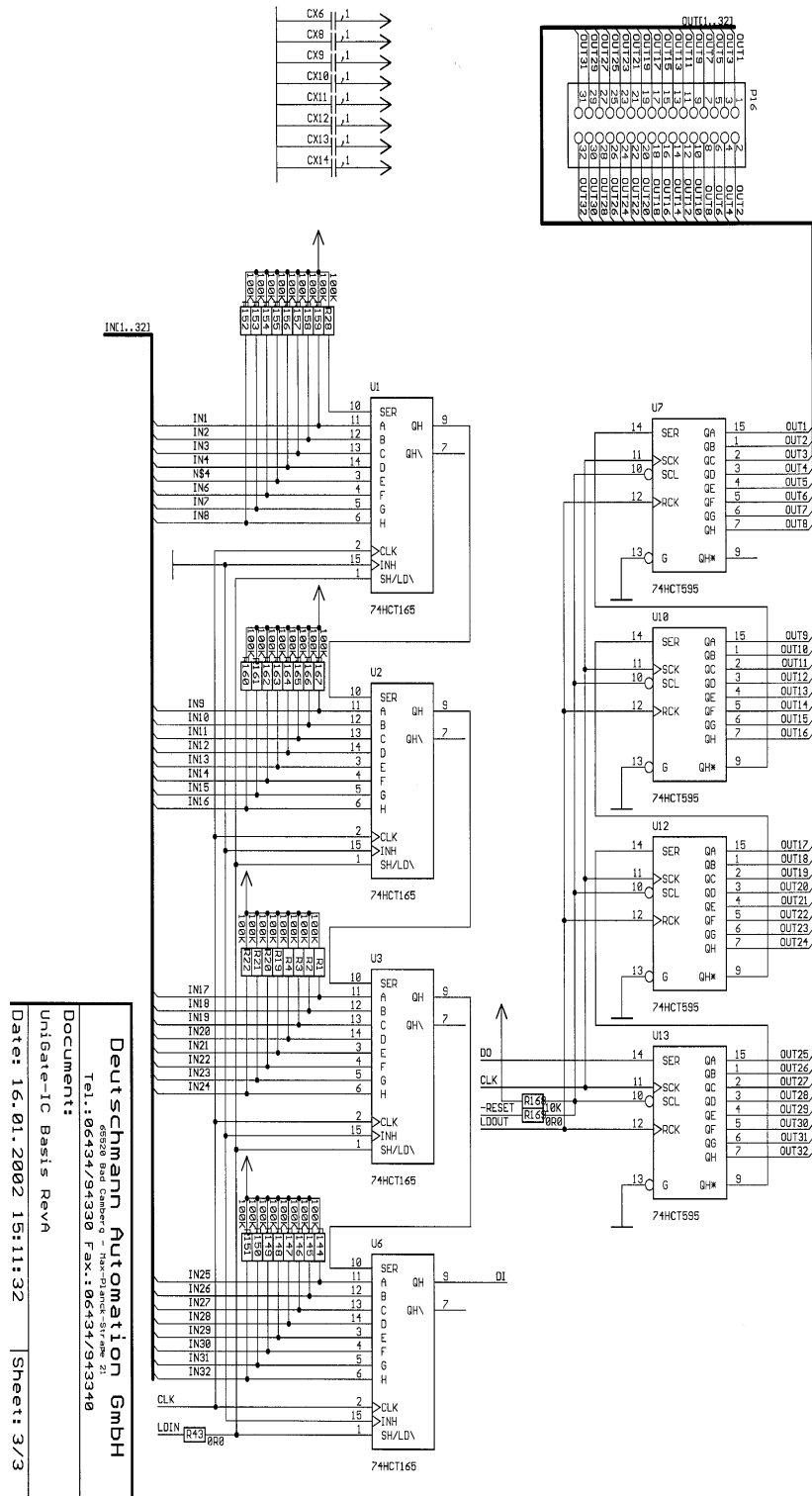
13.4 Debug cable for basis board with UNIGATE® IC

As accessory a configured Debug cable is available. The Debug cable consists of a 9-pin D-SUB socket (basis board RS-side) with two exits:

- a 9-pin D-SUB socket for the connection to Debug COM-PC
- a 5-pin D-SUB Phoenix socket + 2 adapter cables,
one with a 9-pin D-SUB socket for the connection to COM-PC (RS232) and one with open cable ending for the connection to the customer's application (RS232/485).

14 Wiring diagram UNIGATE® IC-basis board DeviceNet





Deutschmann Automation GmbH
 65520 Bad Camberg - Fax-Planck-Str. 21
 Tel.: 06434/94330 Fax.: 06434/94340
 Document:
 Unigate-IC Basis RevA
 Date: 16.01.2002 15:11:32 Sheet: 3/3

15 Servicing

Should questions which are not covered in this manual crop up, please contact us directly.

Please note down the following information before calling:

- Device designation
- Serial number (S/N)
- Article number
- Error number and error description



**1630 W. Diehl Rd.
Naperville, Illinois 60563
+1 630 245-1445, +1 630 245-1717 FAX
www.gridconnect.com**

15.1 Downloading PC software, EDS files and Script examples etc.

You can download the current version of WINGATE[®], a sample EDS file and Script examples free of charge from our Internet server.

Here you will also find the software tool Protocol Developer for UNIGATE[®] SC and IC.

<http://www.deutschmann.de>

