User Manual

HTS

Hengstler Terminal Server

for Windows

Published by:
HENGSTLER GmbH
Postfach 1151
78550 Aldingen

Part number: 2 723 101
Revision date: 10.02.98

We reserve the right to make technical modifications and improvements to our products in line with our policy of continuous improvement.

Contents

Introduction

This manual describes the installation and setup of the HTS. It also contains information about the interfacing methods that are used to communicate with application programs. The contents herein applies to:

- **HTS for Windows 3.x**
  Version: 1.47 or higher
  Article No. 0 723 165
- **HTS for Windows 95 and NT**
  Version: 1.47 or higher
  Article No. 0 723 167

Both of the above mentioned variants behave equally regarding the program execution and handling except of the installation procedure. Not contained are descriptions about the internal data structures of counters, RTC, Booking Terminals or Lamp Position Indicator. Please refer to the device documentation in question.

Tasks of HTS

The following picture gives a functional overview of the tasks HTS performs on the PC platform:

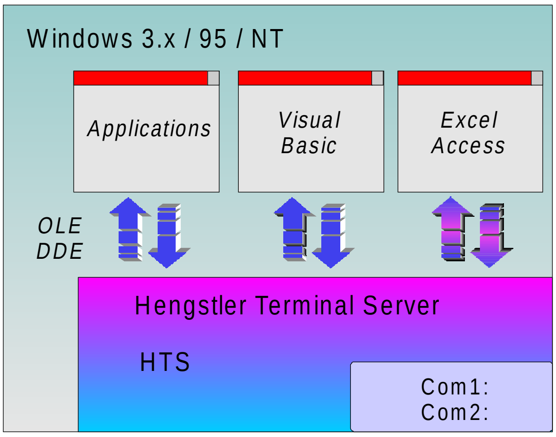


Figure 1: HTS with OLE and DDE

DDE: Dynamic Data Exchange
A so called DDE Server provides for the collection of the variable data (in this case the counter registers). Other applications are called DDE clients; they exchange data with the DDE server via specific mechanisms. Global memory space within Windows is used for the buffering of the data. This global memory space, however, can only be used by 16-bit programs.

OLE: Object Linking and Embedding
With OLE, communication is based on the embedding of functions provided by the OLE server rather than on the mere data. The data required is obtained by means of such functions.

DDE and OLE functionality are both provided at the same time.

Tasks of HTS	Description
Installation Program	• installation procedure • program group and icon
Communications process handling the PC's COMx	• Serves as master for the underlying data protocol *Hengstler TP3* • handles modems • direct communication in case of point-to-point • indirect communication via bus converter RTC
User interface	• configuration of HTS • configuration of the connected devices • statistics functions • diagnostics • errror messages
Application communication interface	• DDE (Dynamic Data Exchange) for 16 bit applications • OLE (Object Linking and Embedding) for 16 and 32 bit applications

Table 1: Tasks of HTS

Compatibility Issues

The following table shows the communication channels between HTS and applications under the different operation systems. Please note that DDE is only defined for the 16 bit address space and therefore is only available for 16 bit applications.

Platform	possible communication
Windows 3.1 / Windows 3.11	OLE (16 bit) DDE (16 bit)
Windows NT / Windows 95	OLE (16 bit or 32 bit) DDE (only for 16 bit programs)

Table 2: HTS communication

Topology

Two basic net topologies are supported:

- **Point-to-point**
  A single counter connects directly to the PC. The counter is equipped with an RS232 interface using TP3 protocol.
- **Network of up to 31 counters**
  The counters are equipped with an RS485 interface using TP3 protocol and are connected to form a 2-wire bus. The bus converter RTC transforms the RS485 bus to the RS232 COM interface of the PC. HTS and RTC are designed to fit to each other perfectly.

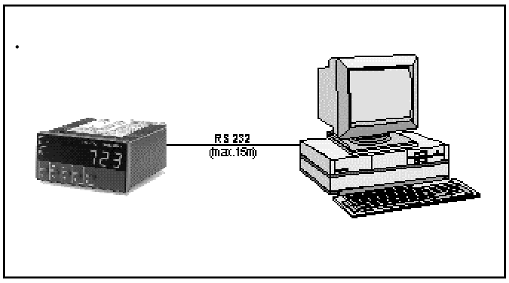


Figure 2: Point-to-point connection

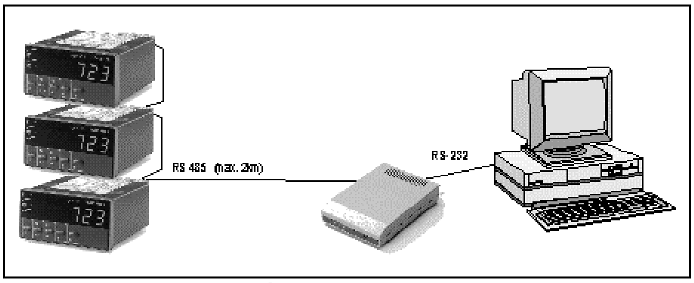


Figure 3: Counters on RS 485 bus

4 basic arrangements are possible on one COM interface of the PC:

S)	**1x Counter**	**RS232**			**<—>**		**RS232**	**COMx**
S1)	**1x Counter**	**RS232**		**Modem**	**<—>**	**Modem**	**RS232**	**COMx**
N)	**Counter 1** **...** **Counter 31**	**RS485** **...** **RS485**	**RTC**		**<—>**		**RS232**	**COMx**
N1)	**Counter 1** **...** **Counter 31**	**RS485** **...** **RS485**	**RTC**	**Modem**	**<—>**	**Modem**	**RS232**	**COMx**

Table 3: Basic arrangements

Installation

System requirements

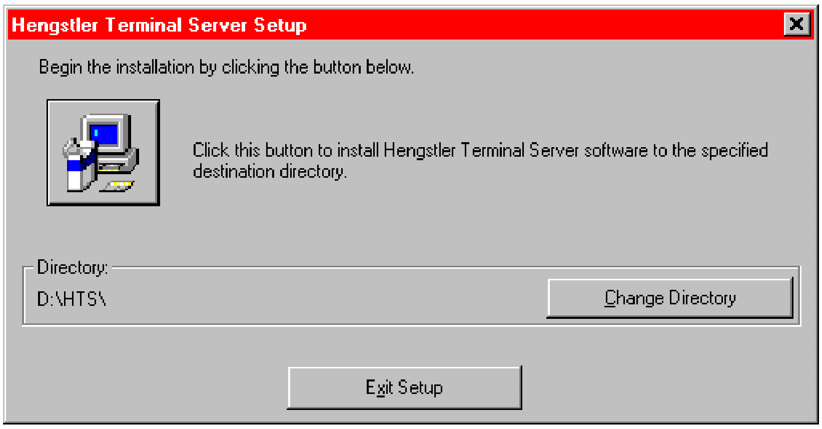
Your PC system should fulfill the following minimum requirements:

- IBM compatible PC
- 386 CPU or higher
- 4 MB memory
- Windows 3.1 / 3.11 or Windows 95 or Windows NT
- one unused COM interface which is configured correctly under Windows
- approx. 2 MB of free disc space

Setup

Insert the installation disc in the floppy drive and start ***setup.exe***. You are requested to change the target directory if you wish to and then start the installation procedure by clicking the big button. The Setup program copies all required files to your PC's hard drive and creates a program group "Hengstler Terminal Server" containing a start symbol.

If you have already used a version of HTS and you want to carry out an update on the same directory, you must delete the executable ***hentmsrv.exe*** prior to installing the new version. Setup does not automatically overwrite the existing file!



Hengstler Terminal Server Setup

Begin the installation by clicking the button below.

Click this button to install Hengstler Terminal Server software to the specified destination directory.

Directory:
D:\HTS\

Setup reports a successful installation with the following dialogue box:

Hengstler Terminal Server Setup

Hengstler Terminal Server Setup was completed successfully.

Autostart

In order to load HTS automatically upon each Windows startup you have to insert the HTS start symbol into the “autostart” program group.

Under Windows 3.1x and NT:
1. open program group "autostart"
2. open program group "Hengstler Terminal Server"
3. arrange both groups so they are visible simultaneously
4. click the program symbol "Hengstler Terminal Server" once and move it into the program group "autostart" while depressing the ctrl key

Under Windows 95:
1. click on START with the right mouse button and then “open”
2. move to the directory "Programs" and then to "Hengstler Terminal Server"
3. click the program symbol "Hengstler Terminal Server" once. Copy it by pressing ctrl-C
4. move back one dirctory level and then into "autostart"
5. add a new copy of the program symbol to this group by pressing ctrl-V

If you do no longer wish HTS to be auto started, delete the program symbol from the autostart program group.

Startup

HTS comes with a predefined configuration which is valid after installation and shown in the table below. If the system topology exactly matches the default configuration and if the counter has been programmed for the physical bus address 1 (see F Codes) then you need not change the HTS setup prior to operation.

We recommend you carry out all steps of setup as described below.

Device	Terminal Number	Terminal Name	physic. Address	Terminal Class	Baud rate	COM Port
Counter	0001	Zaehler1	1	0005-Zähler	SCC1: 4800 Bd	COM1
RTC	0000	RTC	0	0000-RTC	SCC1: 9600 Bd SCC2: 4800 Bd	COM1

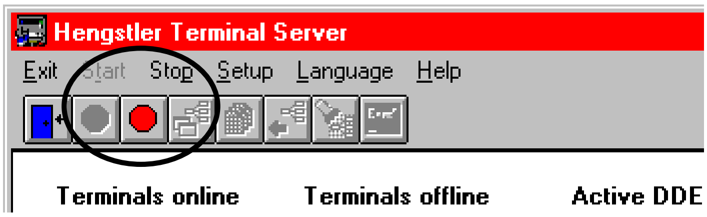
Table 4: Defaul settings of HTS after installation

Disconnect all counters and the RTC from the PC’s COM interfaces.
The startup procedure is as follows:

1. **Primary HTS Startup**
   COM interfaces not yet connected, no counter connected
   OLE capabilities are registered in the Windows registry database automatically
2. **Adjust general driver settings**
   Menue: Setup -> Driver Settings
3. **Configure the counter(s) and the channel(s) according to system topology**
   Determine your system topology
   Menue: Setup -> Terminal Classes
   Menue: Setup -> Terminal Configuration
4. **Starting HTS**
   Menue: Start
   Readout of the main screen’s individual status fields
   Overview of the different settings
   Interpretation of messages and statistics

Primary Startup

In order to adjust all parameter settings HTS is best started without counters or RTC hardware connected. You can start HTS by double clicking on its program icon in the program group or under Windows 95 via Start->programs->Hengstler Terminal Server (throughout the reset of this manual it will no longer be explained how to start HTS).
You may ignore any messages during this startup.

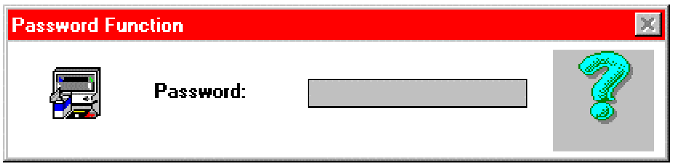


Put HTS offline by clicking the red Stop symbol or the menue *Stop*.

Changes to the settings can only be carried out while HTS is in the Stop state (= offline). If your installation does not come up in **English Language** you may switch to English under the menue Sprache.

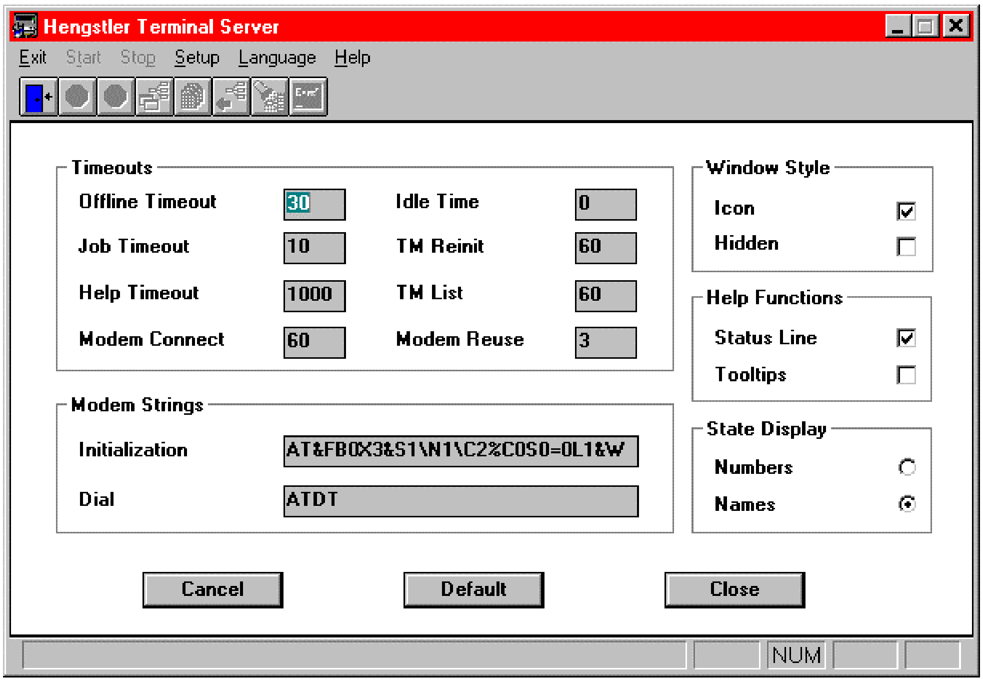
HTS uses the expression **Terminal** (Terminal = terminating node) for all devices that can be connected to HTS, including counters. Any connected unit is regarded as a terminal. **Open** means setting up a connection to a terminal.

When entering the menu *Setup* for the first time in a session you are prompted for the appropriate password which is provided with the distribution set.



Password: **hts1996**

Setup - Driver Settings



Timeouts

Offline Timeout	30	Idle Time	0
Job Timeout	10	TM Reinit	60
Help Timeout	1000	TM List	60
Modem Connect	60	Modem Reuse	3

Modem Strings

Initialization AT&FB0X3&S1\N1\C2%C0S0=0L1&W

Dial ATDT

Window Style

Icon ☑

Hidden □

Help Functions

Status Line ☑

Tooltips □

State Display

Numbers ○

Names ●

The values in ( ) show the default setting.
⇒ marked parameters may be changed. All other parameters should only be touched by experienced staff.

Offline Timeout (30)
Number of polling cycles (sequence within protocol) after which a non responding RTC is reported offline.

Job Timeout (10 seconds)
Maximum time in seconds HTS will be waiting for a response package after a job has been started. If the time elapses an appropriate message is distributed and the job is cancelled.

⇒Help Timeout (1000 milli seconds)
This parameter is the delay time (in ms) after which a yellow tooltip appears when positioning the mouse pointer on a button or menue item. Not effective in this version. Instead a help text appears in the status line.

⇒Modem Connect (60 seconds)
Maximum time (in seconds) a modem connection setup may last. If HTS does not succeed to build up a modem connection after that time the connection process is aborted and a message created.

⇒Idle Time (60 seconds)
HTS terminates automatically after the given time of inactivity of the DDE or OLE server, i.e. after the last DDE or OLE request. An entry of 0 disables the automatic termination. Automatic termination timeout only applies while the Server is online.

TM Reinit (60 seconds)
This time parameter specifies how long HTS waits before re-opening a terminal of which no initialisation data could be received successfully. This parameter does not apply for counters and the RTC.

TM List (60 seconds)
Interval time in seconds after which the RTC shall read the terminal list. The RTC also reads the terminal list each time an appropriate message command is sent. This parameter is not relevant for the counter environment.

⇒Modem Reuse (3 seconds)
After closing a terminal the modem connection is maintained for the time period given by this parameter. This helps to avoid redialing when terminals are opened and closed in short intervals.

⇒Status Line
This parameter determines whether help texts are shown in the footer area of certain masks.

⇒Tooltips
This parameter determines whether help texts are to appear as bubbles and has no effect in this program version (see also under "*Help Timeout*").

⇒Icon
If marked, HTS starts up iconized instead of visible. However, HTS is available in the task list which shows up by pressing the ALT and TAB keys simultaneously.

Hidden
Determines, whether the server runs invisibly in the background. This parameter has priority over the Icon parameter. If hidden, the server does not appear in the Windows' task list.

The marking of this parameter cannot be undone within a mask because there will no longer be a mask available. Also, in this mode, the server can only be terminated by defining an *Idle Time* (see above) or by terminating the Windows session, since the icon is not available.

You can reset the normal mode by editing the INI file ***hentmsrv.ini*** with an appropriate editor and modify or add the line ***WindowMode=0*** under the section ***[Global]***. Please watch the case of the letters.

⇒State Display Numbers / Names
If *Numbers* is checked, the main mask of the server shows the logical Terminal Number in the columns "Terminals online" and "Terminals offline", whereas if *Names* is checked the logical Names of the Counters and RTC are shown as defined in the Terminal Configuration mask.

⇒Initialisation / Dial
The *Initialisation* string is sent to a modem prior to building up a connection (only if a telephone number is given for a Terminal in the Terminal Configuration mask). The string in *Dial* is placed immediately before the telephone number string sent to the modem. Enter the strings which are relevant for your telephone installation and adapt them to the needs of your modem. For details refer to the manual of the modem you will be using. HTS transfers the modem answers in readable text and therefore the init string must contain a V1 (V0 would mean using a binary format). Register S0 specifies the number of rings before the modem answers a call, and must contain a value of 1 or higher. S0=0 means no incoming calls are answered. The DTR control line interrupts the call or accepts it, so you have to insert &D2 into the string. Turn off the echo mode by E0. The number of receipt messages must be limited to 10, which is done with X4. Most modern modems disconnect after a certain period of inactivity which is recommended as an additional safety measurement to avoid high telephone charges in case of a system failure.

The AT commands may vary between the different manufacturers.
Refer to your modem's manual.

Example:
Init string for ***ELSA*** 's modem ***Microlink 2410T2:*** `ATE0V1X0&S0&D2S0=1`

Description:

AT	prefix for AT command
E0	no command echo
V1	answers appear in readable ASCII
X0	do not wait for the ready tone
&D2	DTR goes OFF: modem hangs up, returns to command mode and disallows dialing. DTR goes ON again: dialing is possible.
S0	a value of 1 or higher indicates that the modem answers incoming calls

System Topology

In order to correctly configure the terminal classes and parameters it is essential to understand the assignments within the system. Keep in mind that the RTC has two SCCs (Serial Communication Controller) whereas SCC1 is always connected to the PC and SCC2 serves the counter bus. Please also refer to table 3.
"RS 232" can be substituted by "RS232-Modem - - - Modem-RS232".

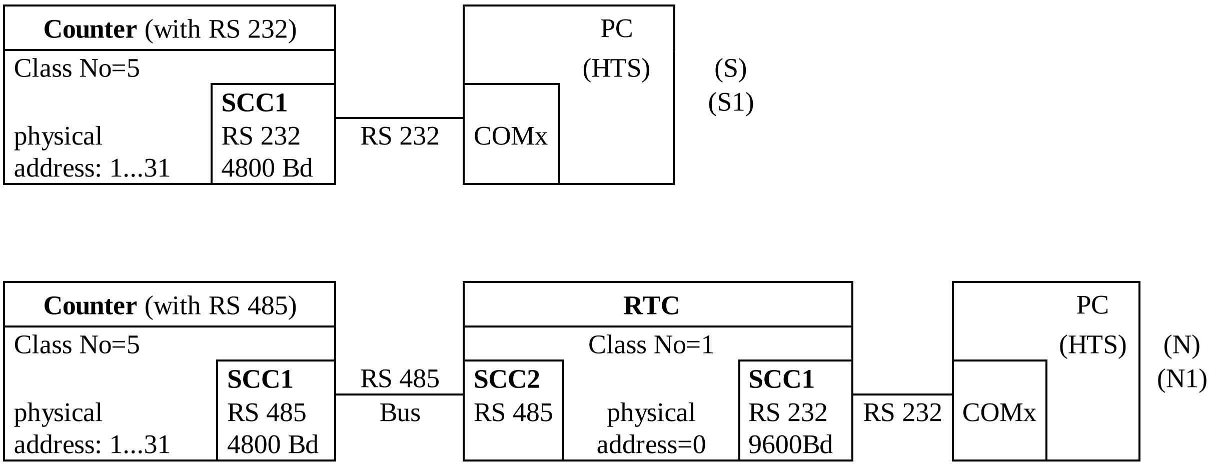


Figure 4: Fixed and variable assignments (= fixed, ... variabel)

SCC1 of class number 5 (hardware type Counter) can currently handle only up to 4800 Bauds. The RTC's SCC2 is preconfigured to 4800 Bauds while its SCC1 is setup for 9600 Bauds.

For each COM interface you will be using, determine the network topology as described in the chapter Introduction/Topology. You have the choice of the following:

- Point-to-point (S)
- Point-to-point via modems (S1)
- 1 to 31 counters via RTC to PC (N)
- 1 to 31 counters via RTC and modems to PC (N1)

Point-to-point (S), (S1)

Menue Terminal Configuration: (Do not forget to save changes!)

1. Determine a Logical Terminal Number for the counter within the range of 1 to 9999 and a Terminal Name. The counter is uniqely identified by this number throughout the whole system.
2. Determine a physical address for the counter within the range 1 to 31. Select the appropriate COM port.
3. Delete the RTC from the list of terminals (select the RTC and press the Delete button). There must be no RTC present in this configuration otherwise the counter will not be served properly.
4. If the counter connects via modem, also enter the complete telephone number under Modem/Network Address (do not enter any non-numeric characters)

Counter Hardware:

5. Check the communication settings of the counter and change the function code for the physical address according to the above point 2. Refer to the counter manual for details how to program the F Codes.

Menue Terminal Classes:

6. Adjust the settings for Baudrate, Data Bits and Parity of SCC1 for Class Number 5 (Counter) according to the counter’s values. Please watch for the maximum baudrate of the counters.

Several Counters via RTC to the PC (N), (N1)

Menue Terminal Configuration: (Do not forget to save changes!)

1. For each of the counters, determine a Logical Terminal Number within the range of 1 to 9999 and a Terminal Name. Each counter is uniqely identified by its logical number throughout the whole system.
2. Determine a physical address for each counter within the range 1 to 31 (and write them down).
3. Check if there is an RTC in the list (Terminal Class is 0001-RTC). If there is no RTC enter a new Terminal Number and select “0001 - RTC” as its Terminal Class. Change its physical address to 0. Use the Store button to save the changes.
4. If the RTC connects to the PC through a modem, also enter the complete telephone number under Modem/Network Address (use only numeric characters).

Counter Hardware:

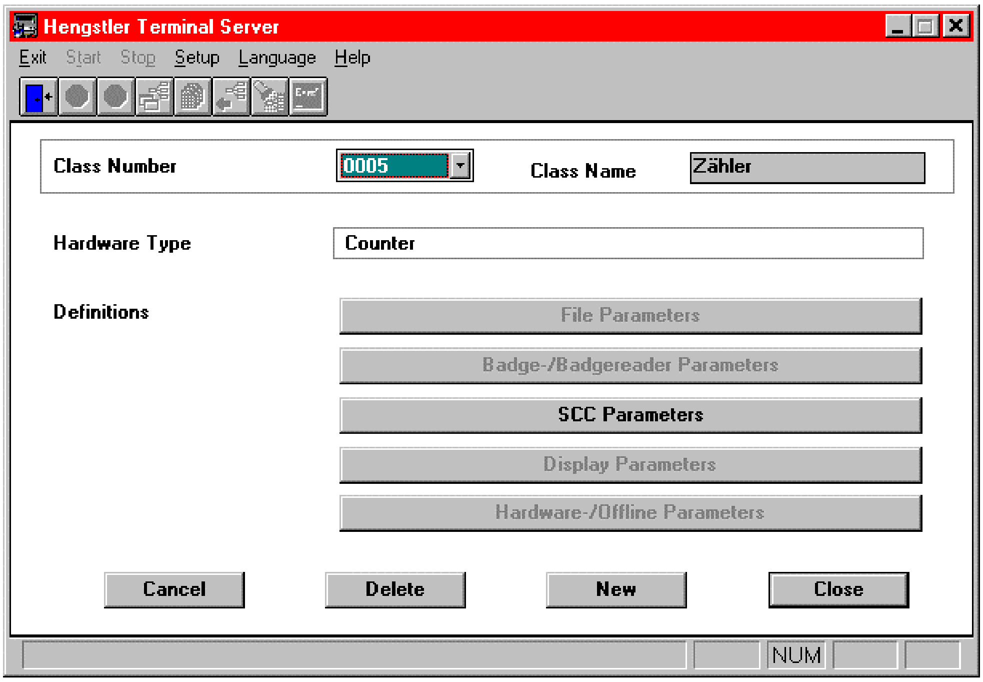
5. Adjust the baudrate, data bits and parity settings of each counter. It will be same for all counter. Change the physical address of each counter according to your predeterminations (see step 2). Refer to the counter manuals for programming of function codes.

Menue: Terminal Classes:

6. Adjust the settings for baudrate, data bits and parity of SCC1 for Class Number 5 (Counter) according to the counters’ values. Please watch for the maximum baudrate of the counters.
7. The settings for baudrate, data bits and parity must be the same as for the RTC’s SCC2 interface.
8. Set the baudrate for SCC1 of Class 1 (RTC) to its maximum value.

Terminal Class Definitions

The server must be stopped before you can start this mask in the Setup menue. All shaded fields are not applicable for the device Classes in question and cannot be activated.

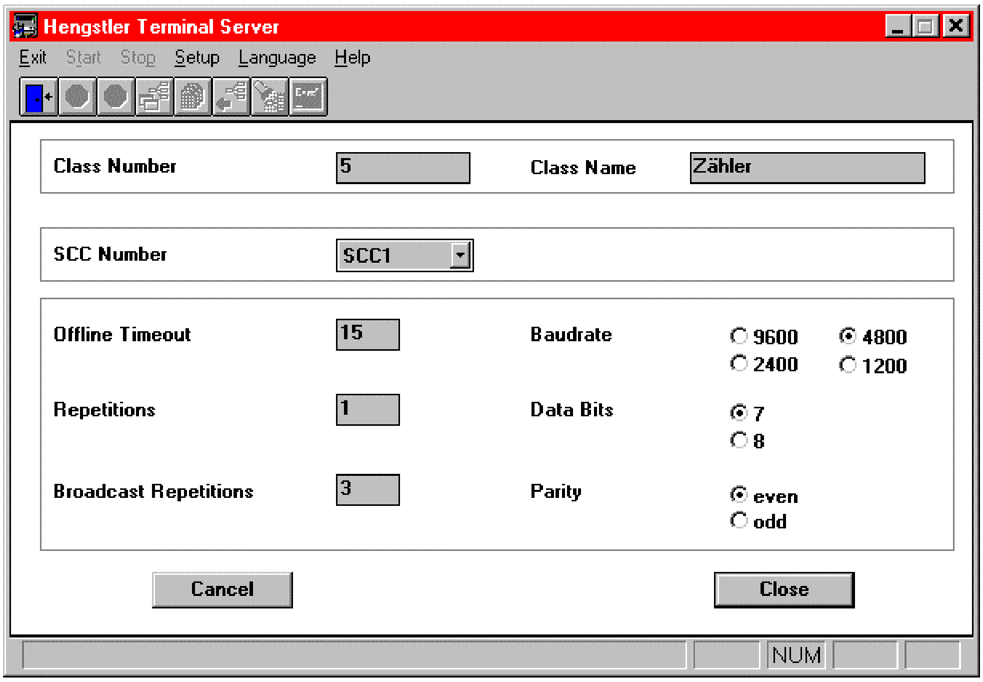

Hengstler Terminal Server
Exit Start Stop Setup Language Help
File Parameters
Badge-/Badgereader Parameters
SCC Parameters
Display Parameters
Hardware-/Offline Parameters
Cancel
Delete
New
Close
NUM

Class Number: 0005
Class Name: Zähler
Hardware Type: Counter
Definitions

Class Number
The referential number of the class is assigned internally and cannot be changed.

Class Name
The referential name of the class is predefined in German language and may be changed for the adaption to another language (in the above example “Zähler” would translate to “Counter”).

SCC Parameters
This button leads you to the mask with the serial interface(s) parameters of the selected Terminal Class. It looks like:


Hengstler Terminal Server
Exit Start Stop Setup Language Help
Cancel
Close
NUM

Class Number: 5
Class Name: Zähler
SCC Number: SCC1
Offline Timeout: 15
Repetitions: 1
Broadcast Repetitions: 3
Baudrate: ○ 9600 ● 4800 ○ 2400 ○ 1200
Data Bits: ● 7 ○ 8
Parity: ● even ○ odd

SCC Number
Counters have only a serial interface and thus offer only a single choice.

Offline Timeout
Determines the Offline Timeout of a Terminal (seconds). If a Terminal out of the terminal list does not respond within this time period after it has been requested for it will be marked offline.

Repetitions
This parameter sets the number data transmission is repeated after an erroneous transmission sequence.

Broadcast Repetitions
This parameter sets the number a of repetitions for broadcast messages during the inter-terminal communication and is not relevant for counters nor the RTC.

Baudrate
Sets the data transfer rate. Please watch for the baudrate supported by the device classes.

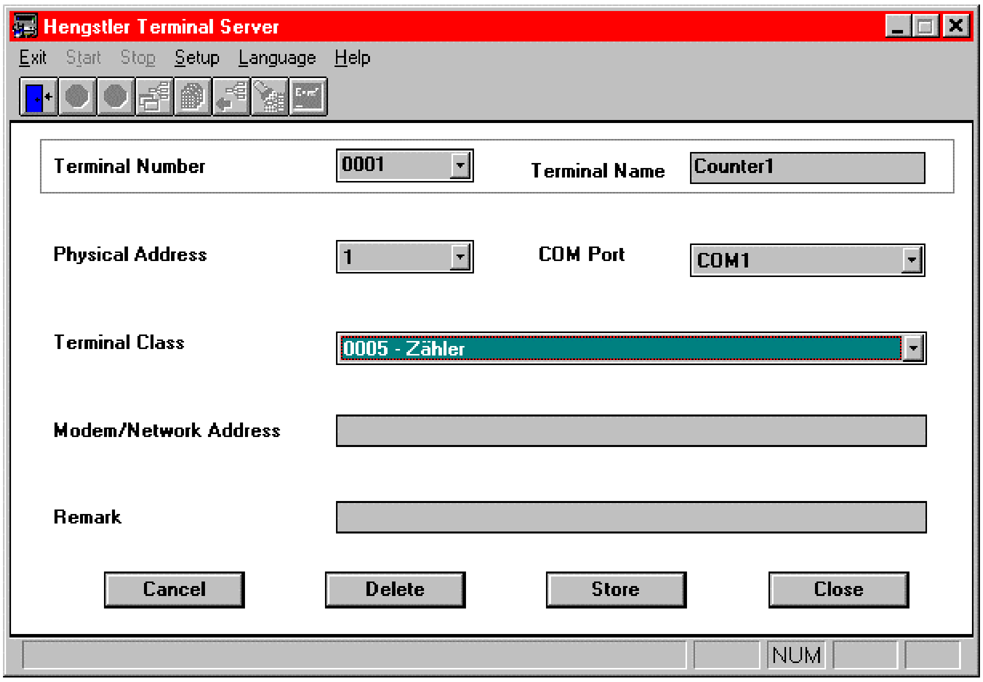
The bus interface SCC2 for the RTC is by error sometimes set to 9600 when the mask is entered. Before you leave the mask, the value must be set to 4800 which is the highest possible baudrate for counters.

Data Bits / Parity
The number of data bits should always remain set to 7 and even parity and must match the settings of the appropriate function codes of the counters.

Terminal Configurations

The server must be stopped before you can start this mask in the Setup menue.

ℹ Don't forget to store the new settings before you quit the mask.


Hengstler Terminal Server
Exit Start Stop Setup Language Help
Cancel
Delete
Store
Close
NUM

Terminal Number: 0001
Terminal Name: Counter1
Physical Address: 1
COM Port: COM1
Terminal Class: 0005 - Zähler
Modem/Network Address:
Remark:

Terminal Number
This field selects the logical number of the available Terminals. A new Terminal entry is created by typing in a new number in the range of 1 to 9999.

Terminal Name
The Terminal Name is associated with a Terminal Number and is not checked for uniqueness. It is displayed in the online or offline column if the field “State Display” is set to Names in the “Driver Settings” mask (see above).

Physical Address
The Physical Address of the Terminal. Each counter connected to the same bus must have a unique entry in the range of 1 to 31. The RTC must always use the physical address 0.

COM Port
Specifies the PC's serial port on which this Terminal can be reached. This field corresponds to COM**x** in Figure 4.

Terminal Class
This field determines the hadware type (device class) of the Terminal (Counter or RTC).

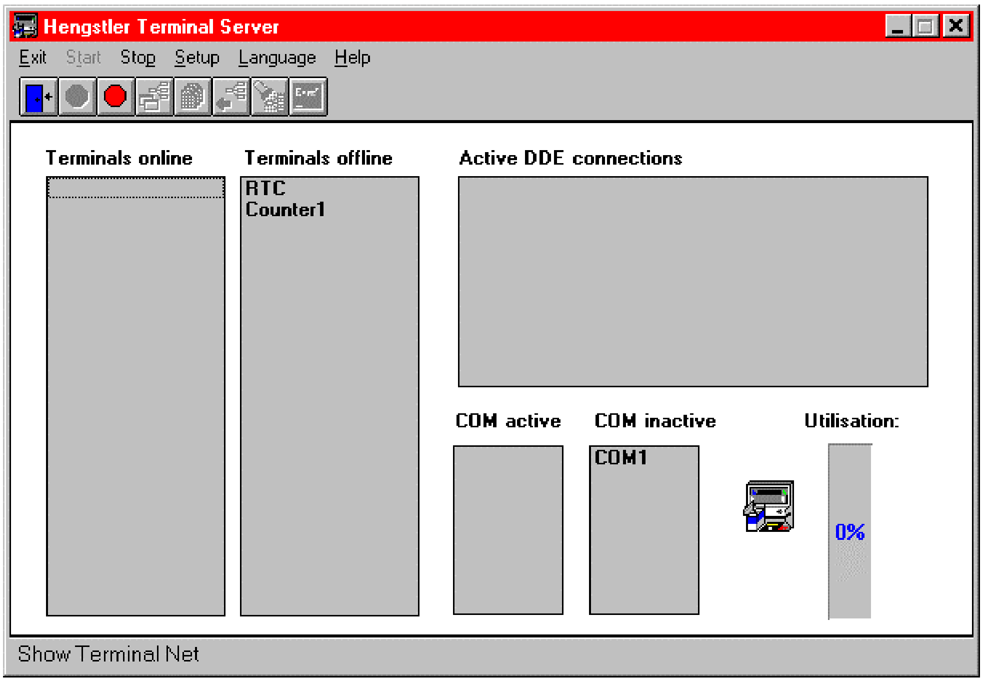
Modem/Network Address
Enter a telephone number if your system configuration uses modems. See also under **Initialization** and **Dial** in the “Driver Settings” mask. Do not use any special characters other than digits.

Remark
You may use this field as a comment field assigned to this Terminal. It is not used by the server.

Starting HTS

After you have finished all settings you can connect the hardware. Make sure that the length of the RS 232 cable is within the allowed range. If you use an RS 485 bus on the RTC please check whether a bus termination resistor is connected to the outermost counter (see the counter manual or supplementary manual).

Start the server by restarting the program or by clicking the green start button or via the Start menue.



This screen appears after startup and gives you an overview of the server's operating status.

Terminals online
This window shows all Terminals which are configured properly and have an online connection to the server.

Terminals offline
This window shows those Terminals which are configured properly but not having a valid connection to the server (e.g. modem connected Terminals with no dial connection).

When you double click on a Terminal entry an overview window opens showing statistical data of the appropriate Terminal. For details refer to the chapter Diagnostics.

COM active
This column lists all those COM ports on which at least one valid Terminal (RTC or Counter) has been detected.

COM inactive
This column lists all those COM ports on which no valid Terminal (RTC or Counter) has been detected. Normally, this column should not contain any entries.

When you double click on any of the entries, a line monitor window appears showing the data stream of the appropriate COM port. For details refer to the chapter Diagnostics.

Active DDE connections

This window shows all window handles (an internal sequence number by the Windows operating system for every application window) and window titles of those custom applications which have submitted a valid DDE connection to the server. Each new incoming OLE request gets a new sequence number which appears in this window until that OLE request has been served.

Utilisation

A graph shows the current utilisation of the server program compared to the maximum possible capacity.

Integration into Customer Application

Communication to customer applications is carried out with the mechanisms provided by DDE and OLE.

DDE

HTS has been implemented as server providing communication services for client programs. All of the inter-process communication to this server runs asynchronously, because it may take a certain time until the server has collected all the requested data via the serial interfaces.
For reasons of realtime behaviour, processor utilisation and portability to non-windows systems this server uses a custom DDE protocol as described herein.

Communication Mechanism

There is always a fixed handshake during communication. Once the DDE request is accepted by the server all timeout control and error handling is done by the server program. The client application receives either the result of the DDE request (with or without data) or a qualified error message. Net data are delivered within the acknowledgement message.
DDE servers offer their services under certain names which the client program must know in order to select the appropriate service.

Service: "Hengstler Terminal Server"
Topic: "All Items"
The data exchange is always asynchronous and binary. The command structure must be used for DATA requests.

Schematical Overview: Server Communication

Application		HTS
DDE Connect	⇒	Accept the client
Connection to HTS setup	⇐	DDE Acknowledge
DDE Data Request	command structure ⇒	communicating with counter in question.. Read or write data from/to counter
process net data	⇐ command structure	DDE Data
DDE Disconnect	⇒	Remove Client from client list
Connection to HTS cancelled	⇐	DDE Disconnect

Command Structure

The command structure which HTS uses is defined as a structure in C as follows:

```
typedef struct tp3_command_s {
   long size;              // size of structure
   long pid;               // window handle of application
   long logtnr;            // logical terminal number (0000-9999)
   long jobid;             // requests counter
   long service;           // service (see below)
   long iodesc;            // File/Device number for Input / Output
   long datatype;          // reserved
   long records;           // number of Records
   long response;          // with/without acknowledgement
   long error;             // error
   char errortext[ERRTEXTLEN]; // error type
   char filename[MAXPATH+1];  // file name
}tp3_command_struct;
```

The command structure is the header of each DDE data package. If there is data provided with a DDE package, the data is appended directly to the command structure and the size parameter represents the increased total length value.

Note: Within DDE the received data structure is always deallocated by the receiver of the message!

The existance of data can always be identified by the **size** paramter of the command structure.

number of databytes = ***size*** *- sizeof(tp3_command_s);*

size
Contains the total number of bytes of the DDE package including additional parameters or data. This is the allocated total size of the package. size must always be filled in.

pid - Process ID
Must always contain the window handle of the client application.

logtnr - Logical Terminal Number
Throughout all communication the counters and RTC are identified by their logical terminal number as defined in the Terminal Configuration menu. HTS translates each logical terminal number into a physical address together with a COM Port number and the communication type (direct, modem). Must always be filled in.

jobid - request counter
The value of this sequential counter increments for each new request and goes with the request. HTS enters this value in the acknowledgement package. Optional value.

service
Describes the requested service. Please use the definitions of the header file ***types.h***. Must always be filled in by the client.

iodesc - file / service number
This parameter describes the file descriptor or in case of a device command the number of the device which should be processed. It depends on the requested service. The definitions can be found in ***types.h***. For device commands the following rule applies: iodesc = register number + 1000*devcice number.

datatype
reserved, not used.

records - number of records
Contains the number of data records and depends on the service.

response - type of answer
Classifies the package type and can contain the following values (see also ***types.h***):

```
NO_RESPONSE     -> Do not send an answer to this request
SEND_RESPONSE   -> acknowledgement package must be sent
IS_RESPONSE     -> this package is an acknowledgement package
```

Must always be filled in.

error - Return value
If different from NOERROR this parameter specifies the tpye of error that may have occured during the processing of a request. For a detailled description of the individual values please refer to the header files. For a request package error must be set 0. In an acknowledgement package, NOERROR indicates an error free sequence.

errortext - message text
Only used by acknowledgement packages. Contains a brief error description in the language configured in the HTS main menu. Not used by request packages.

filename
reserved, not used.

Data Exchange with Counters

Counters allow read and write access only via so-called devices. There is a predetermined range of numbers assigned to the different hardware types, as followes:

The device number for a counter is **73**.
Eight different registers are possible:

No.	description	accesss
000	Counter Value	(rd, wr, rd&cl)
001	Preset 1	(rd, wr, rd&cl)
002	Preset 2	(rd, wr, rd&cl)
003	Chain Value	(rd, rd&cl)
004	(Re)Set Value	(rd, wr, rd&cl)
005	Prescaler	(rd, wr)
006	Decimal Point	(rd, wr)
007	Version	(rd)

The parameter ***iodesc*** in the command structure calculates as follows:
iodesc = 1000*device number + register number

For a counter (device 73) this results in the following list of possible ***iodesc*** values:

service	counter register	Reg. No.	***service***	***iodesc***	data format* (always ASCII)
Read	Count value	000	READ_DEVICE	73000	8 characters, left aligned
Read	Preset 1	001	READ_DEVICE	73001	8 characters, left aligned
Read	Preset 2	002	READ_DEVICE	73002	8 characters, left aligned
Read	Chain value	003	READ_DEVICE	73003	8 characters, left aligned
Read	Set value	004	READ_DEVICE	73004	8 characters, left aligned
Read	Prescaler	005	READ_DEVICE	73005	8 characters, left aligned
Read	Decimal Point	006	READ_DEVICE	73006	2 ch., left aligned, 0=no dec.pt.
Read	Version	007	READ_DEVICE	73007	8 characters, Text
Read&Clear	Count value	000	RD_CL_DEVICE	73000	8 characters, left aligned
Read&Clear	Preset 1	001	RD_CL_DEVICE	73001	8 characters, left aligned
Read&Clear	Preset 2	002	RD_CL_DEVICE	73002	8 characters, left aligned
Read&Clear	Chain value	003	RD_CL_DEVICE	73003	8 characters, left aligned
Read&Clear	Set value	004	RD_CL_DEVICE	73004	8 characters, left aligned
Write	Count value	000	WRITE_DEVICE	73000	8 characters, left aligned
Write	Preset 1	001	WRITE_DEVICE	73001	8 characters, left aligned
Write	Preset 2	002	WRITE_DEVICE	73002	8 characters, left aligned
Write	Set value	004	WRITE_DEVICE	73004	8 characters, left aligned
Write	Prescaler	005	WRITE_DEVICE	73005	8 characters, left aligned
Write	Decimal Point	006	WRITE_DEVICE	73006	2 ch., left aligned, 0=no dec.pt.

* filled with blanks to the right

Example: writing the value "1234.98" into the count register. If the current decimal point setting is 1, (i.e. xxxxx.x) the counter will show “1234.9” after the job has been carried out.

The data, as described in the above table, is appended directly to the command structure. Please notice that the strings are not zero terminated. The value of ***size*** represents exactly the total number of bytes (command structure + allocated data area). HTS does not carry out any format or length checking. Illegal formatting in a request leads to a negative acknowledgement by the counter, accompanied by an appropriate ***error*** number.

Data strings are not zero terminated!

The Service Read & Clear Count Value has the effect of externally or manually resetting the counter, i.e. all output signal are deactivated if they ara are active and programmed for bistable output.

OLE

HTS displays each OLE request currently under service in the main window’s column “active DDE connections”. Each new request gets a new sequential number. The examples below show you can implement the OLE interface into your application.

Excel and Visual Basic

The most easy implementation is offered by Visual Basic which belongs to Microsoft Excel. In the example the count value is read from the counter with the logical terminal number 25 and its preset_1 register is set to a value taken from a table cell.

```
' Logical Counter address
Konst CounterAddress = 25
' Registers of counter
Konst CountValue = 0
Konst Preset1 = 1
Konst Preset2 = 2
Konst Chain = 3
' Buffer variable
Dim Buffer As Single

' Counter value is read an written into table
Sub Read_Counter()
    Set Hts = GetObject(Class:="Hengstler.TerminalServer.10")
    Result = Hts.ReadRegister(CounterAddress; CountValue)
    Sheets("Table1").Cells(6; 2).Value = Result
End Sub

Sub Write_Preset()
    Buffer = Sheets("Table1").Cells(2; 2).Value
    Set Hts = GetObject(Class:="Hengstler.TerminalServer.10")
    Result = Hts.WriteRegister((CounterAddress; Preset1; Buffer)
End Sub

Sub Read_Clear_Counter()
    Set Hts = GetObject(Class:="Hengstler.TerminalServer.10")
    Result = Hts.ClearRegister(CounterAddress; CountValue)
    Sheets("Table1").Cells(6; 2).Value = Result
End Sub
```

Example: OLE with Visual Basic

C++ Applications

The following example is based on ***MS Visual C++ Design Studio***. It is taken from the sample application which can be found in **`<installation path of MSVC>\samples\ole\calcdriv`**.

The following procedure steps automatically generates the class definitions for "Hengstler Terminal Server":

1. Terminate HTS
2. On the command line, run **hentmsrv.exe /typelib**
   This creates an object library file ***hentmsrv.olb*** in the installation directory of your HTS
3. With the help of the Class Wizard, create a new class deriving from ***hentmsrv.olb***.
4. the ***class HengstlerTerminalServer*** is ready to be used.

Extracts of the different program files visualize the most important passages.

hentmsrv.h:

```
// Machine generated IDispatch wrapper class(es) created with ClassWizard
///////////////////////////////////////////////////////////////////////////
// HengstlerTerminalServer wrapper class

class HengstlerTerminalServer : public COleDispatchDriver
{
public:
      HengstlerTerminalServer() {}       // Calls COleDispatchDriver default constructor
      HengstlerTerminalServer(LPDISPATCH pDispatch) : COleDispatchDriver(pDispatch) {}
      HengstlerTerminalServer(const HengstlerTerminalServer& dispatchSrc) :
COleDispatchDriver(dispatchSrc) {}

// Attributes
public:

// Operations
public:
      CString ReadRegister(long File, long Number);
      CString ClearRegister(long File, long Number);
      CString WriteRegister(long File, long Number, LPCTSTR Value);
      CString ReadDevice(long File, long Number);
      CString WriteDevice(long File, long Number, LPCTSTR Value);
      CString ReadFile(long File, long Number);
      CString WriteFile(long File, long Number, LPCTSTR Value);
};
```

Initialising OLE:

```
BOOL CCalcDrivApp::InitInstance()
{
#ifdef _DEBUG
    #ifndef _MAC
          // turn on extra memory tracking
      afxMemDF |= checkAlwaysMemDF;
    #endif
#endif
      // Initialize OLE 2.0 libraries
      if (!AfxOleInit())
      {
            AfxMessageBox(IDP_OLE_INIT_FAILED);
            return FALSE;
      }

      // Standard initialization
      // If you are not using these features and wish to reduce the size
      //  of your final executable, you should remove from the following
      //  the specific initialization routines you do not need.

      Enable3dControls();    // Use 3d controls in dialogs

      // Simple application that simply invokes a dialog
      CDriverDlg dlg;
      m_pMainWnd = &dlg;
      dlg.DoModal();
      return FALSE;   // don't run after the dialog is done
}
```

hentmsrv.cpp:

```
// Machine generated IDispatch wrapper class(es) created with ClassWizard

// To redo creation create the HTS-OLE typelib by calling
// HENTMSRV.EXE /TYPELIB
// The file HENTMSRV.OLB is automatically generated for you
// In the class wizard choose Add Class and From an OLE TypeLib
// A dialog box appears: Choose the previously generated HENTMSRV.OLB
// This file and the header file will then automatically be generated for you
// You can then directly use all HengstlerTerminalServer member functions

#include "stdafx.h"
#include "hentmsrv.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif



/////////////////////////////////////////////////////////////////////////////
// HengstlerTerminalServer properties

/////////////////////////////////////////////////////////////////////////////
// HengstlerTerminalServer operations

CString HengstlerTerminalServer::ReadRegister(long File, long Number)
{
      CString result;
      static BYTE parms[] =
            VTS_I4 VTS_I4;
      InvokeHelper(0x1, DISPATCH_METHOD, VT_BSTR, (void*)&result, parms,
            File, Number);
      return result;
}

CString HengstlerTerminalServer::ClearRegister(long File, long Number)
{
      CString result;
      static BYTE parms[] =
            VTS_I4 VTS_I4;
      InvokeHelper(0x1, DISPATCH_METHOD, VT_BSTR, (void*)&result, parms,
            File, Number);
      return result;
}


CString HengstlerTerminalServer::WriteRegister(long File, long Number, LPCTSTR Value)
{
      CString result;
      static BYTE parms[] =
            VTS_I4 VTS_I4 VTS_BSTR;
      InvokeHelper(0x2, DISPATCH_METHOD, VT_BSTR, (void*)&result, parms,
            File, Number, Value);
      return result;
}

CString HengstlerTerminalServer::ReadDevice(long File, long Number)
{
      CString result;
      static BYTE parms[] =
            VTS_I4 VTS_I4;
      InvokeHelper(0x3, DISPATCH_METHOD, VT_BSTR, (void*)&result, parms,
            File, Number);
      return result;
}

CString HengstlerTerminalServer::WriteDevice(long File, long Number, LPCTSTR Value)
{
      CString result;
      static BYTE parms[] =
            VTS_I4 VTS_I4 VTS_BSTR;
      InvokeHelper(0x4, DISPATCH_METHOD, VT_BSTR, (void*)&result, parms,
            File, Number, Value);
      return result;
}

CString HengstlerTerminalServer::ReadFile(long File, long Number)
```

```
{
      CString result;
      static BYTE parms[] =
            VTS_I4 VTS_I4;
      InvokeHelper(0x5, DISPATCH_METHOD, VT_BSTR, (void*)&result, parms,
            File, Number);
      return result;
}

CString HengstlerTerminalServer::WriteFile(long File, long Number, LPCTSTR Value)
{
      CString result;
      static BYTE parms[] =
            VTS_I4 VTS_I4 VTS_BSTR;
      InvokeHelper(0x6, DISPATCH_METHOD, VT_BSTR, (void*)&result, parms,
            File, Number, Value);
      return result;
}
```

hconnect.cpp:

```
// This example is taken from the Microsoft example CALCDRIV
// and has been modified for use with Hengstler Terminal Server
BOOL CDriverDlg::OnInitDialog()
{
      CDialog::OnInitDialog();

      // create the hts object that we'll drive through OLE automation
      COleException e;
      CLSID clsid;
      if (CLSIDFromProgID(OLESTR("Hengstler.TerminalServer.10"), &clsid) !=
NOERROR)
      {
            AfxMessageBox(IDP_UNABLE_TO_CREATE);
            EndDialog(IDABORT);
            return FALSE;
      }

      // try to get the active hts before creating a new one
      LPUNKNOWN lpUnk;
      LPDISPATCH lpDispatch;
      if (GetActiveObject(clsid, NULL, &lpUnk) == NOERROR)
      {
            HRESULT hr = lpUnk->QueryInterface(IID_IDispatch,
                  (LPVOID*)&lpDispatch);
            lpUnk->Release();
            if (hr == NOERROR)
                  m_hts.AttachDispatch(lpDispatch, TRUE);
      }

      // if not dispatch ptr attached yet, need to create one
      if (m_hts.m_lpDispatch == NULL &&
            !m_hts.CreateDispatch(clsid, &e))
      {
            AfxMessageBox(IDP_UNABLE_TO_CREATE);
            EndDialog(IDABORT);
            return FALSE;
      }

      // read a counter register to test and see if it works
      CString result = m_hts.ReadRegister(1,1);
      // examine result in the debugger
      return TRUE;  // return TRUE  unless you set the focus to a control
}
```

Tuning

In order to suppress the overhead inside HTS which occurs during open device and close device operations upon each DDE or OLE request a trick can be applied. Append the following lines to the file hentmsrv.ini (after terminating HTS):

hentmsrv.ini:

```
[MultiTasking]
DirectAccess=1
```

This modification should only be carried out if HTS is used exclusively by one application. Please watch the case of the letters.

Diagnostics and Error Handling

This chapter mainly deals with items of servicing.

Settings Overview

This section is intended to help service people under circumstances which require a more comprehensive overview of HTS files. Do not modify or delete INI files as they are manipulated by HTS.

hentmsrv.ini:	general driver settings
vt3tm.ini:	list of all preconfigured terminals
vt3c0000.ini:	definitions of the terminal class 0 (RTC)
vt3c0005.ini:	definitions of the terminal class 5 (Counter)

STOP Manual changes in the INI files are not allowed and may result in unpredictable behaviour. The definitions are case sensitive.

hentmsrv.ini

```
[Global]
Version=1
HelpLevel=2
Language=0
WindowMode=0
LogfileSize=300000
StatusMode=0

[Modem]
InitString=AT&FB0X3&S1\N1\C2%C0S0=0L1&W0Z0
RInitString=AT&F5E0M0X3&R1&S1\N1\C2%C0S0=1&W0Z0
DialPrefix=ATDT

[Timeouts]
HelpTimeout=1000
OfflineTimeout=30
JobTimeout=10
IdleTimeout=0
TerminalListTimeout=60
ReinitTimeout=60
ModemConnectTimeout=60
ModemReuseTimeout=1
ItsByteTimeout=3
ItsEnqTimeout=20
ItsCmdTimeout=4

[COM1]
Device=0
[COM2]
Device=0
...
[COM9]
Device=0
```

vt3tm.ini

```
// Terminal-Definition
// Created by Hengstler Terminal Server
// Last Change: 28.05.1997 18:19:05

[TM0000]
Name=RTC
PhysicalTerminalAdress=0
TMClass=0
COM-Port=1
RemoteAccess=
Remark=

[TM0001]
Name=Zaehler1
PhysicalTerminalAdress=1
TMClass=5
COM-Port=1
RemoteAccess=
Remark=
```

vt3c0000.ini

Class definitions for RTC, terminal class 0

```
// Class-Definition
// Created by Hengstler Terminal Server
// Last Change: 28.10.1996 15:25:24

[Version]
Version=1

[Device]
Name=RTC
DeviceClass=3
DeviceType=3

[SCC1]
Baudrate=5
DataBitsCount=7
Parity=0
OfflineTimeout=5
Repetitions=3
BroadcastRepetitions=2
SendBeginDelay=0
SendEndDelay=0

[SCC2]
Baudrate=4
DataBitsCount=7
Parity=0
OfflineTimeout=5
Repetitions=3
BroadcastRepetitions=2
SendBeginDelay=0
SendEndDelay=0
```

vt3c0005.ini

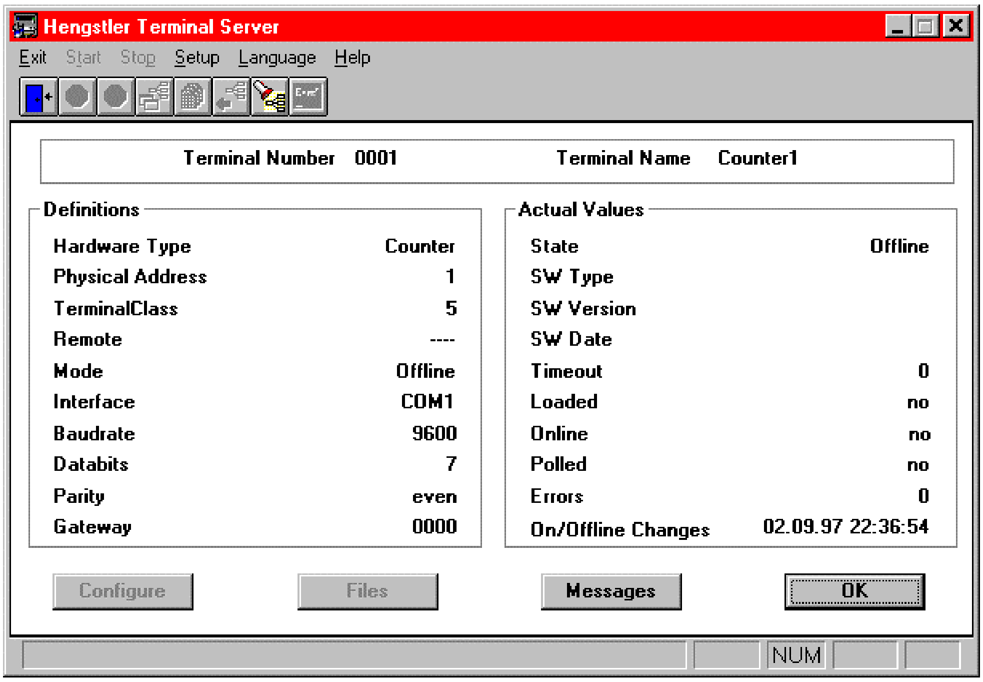
Class definitions for Counter, terminal class 5

```
// Class-Definition
// Created by Hengstler Terminal Server
// Last Change: 03.02.1997 15:15:10

[Version]
Version=1

[Device]
Name=Counter
DeviceClass=1
DeviceType=5

[SCC1]
Baudrate=4
DataBitsCount=7
Parity=0
OfflineTimeout=15
Repetitions=1
BroadcastRepetitions=3
SendBeginDelay=0
SendEndDelay=0
```

Statistics

By double clicking on a terminal entry in the columns *Terminals online* or *Terminals offline* of HTS's main window you get an overview of the device in question. The individual masks are not subject to detailled instructions.

The entry COMx in the parameter line "Interface" always means the PC's COM port which leads to the device either directly or indirectly (via RTC).



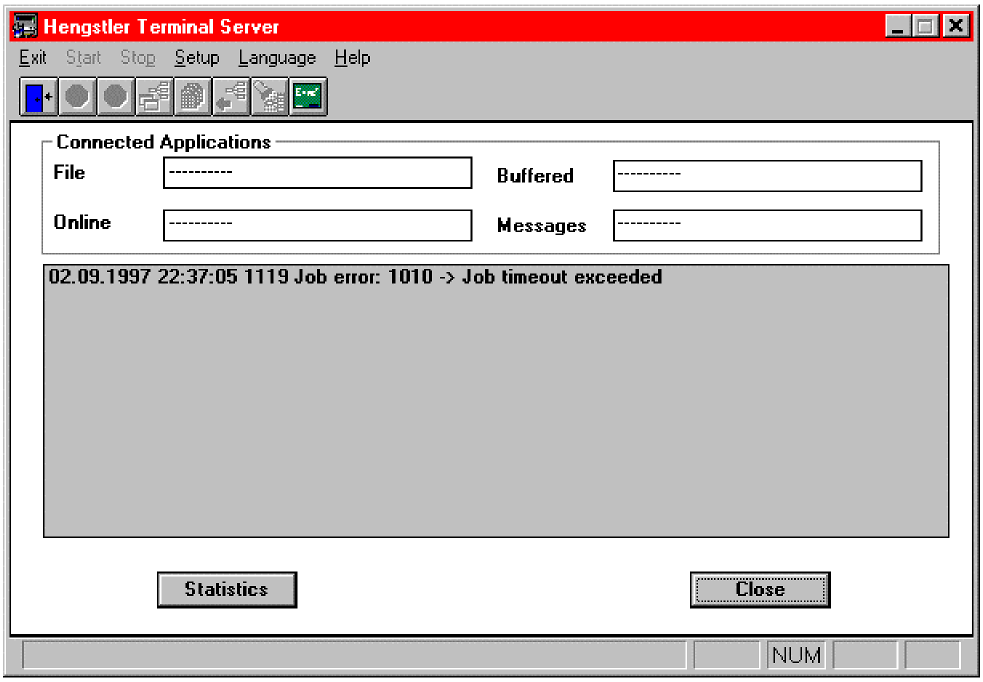
Terminal Number 0001 Terminal Name Counter1

Definitions

Hardware Type	Counter
Physical Address	1
TerminalClass	5
Remote	----
Mode	Offline
Interface	COM1
Baudrate	9600
Databits	7
Parity	even
Gateway	0000

Actual Values

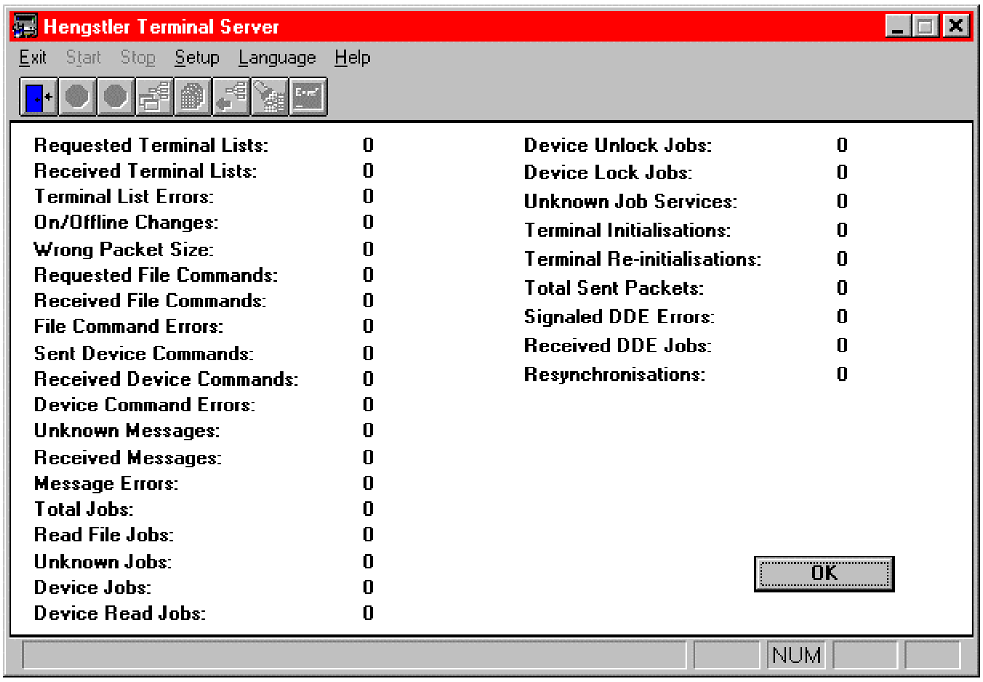
State	Offline
SW Type	
SW Version	
SW Date	
Timeout	0
Loaded	no
Online	no
Polled	no
Errors	0
On/Offline Changes	02.09.97 22:36:54



Connected Applications

File	----------	Buffered	----------
Online	----------	Messages	----------

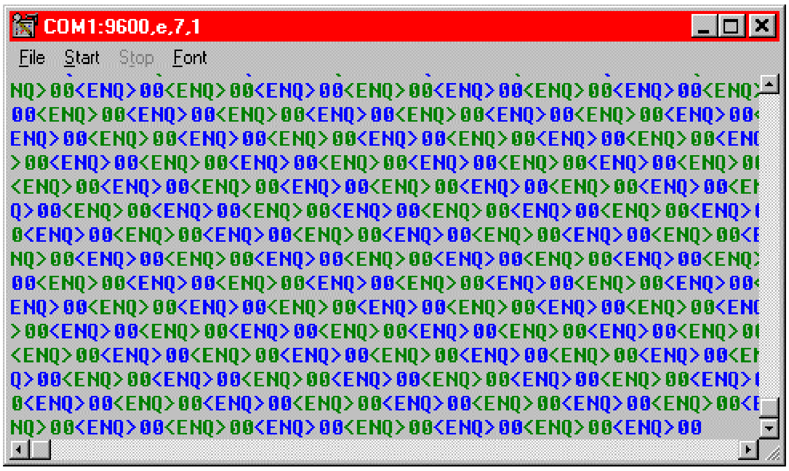
```
02.09.1997 22:37:05 1119 Job error: 1010 -> Job timeout exceeded
```



Requested Terminal Lists:	0	Device Unlock Jobs:	0
Received Terminal Lists:	0	Device Lock Jobs:	0
Terminal List Errors:	0	Unknown Job Services:	0
On/Offline Changes:	0	Terminal Initialisations:	0
Wrong Packet Size:	0	Terminal Re-initialisations:	0
Requested File Commands:	0	Total Sent Packets:	0
Received File Commands:	0	Signaled DDE Errors:	0
File Command Errors:	0	Received DDE Jobs:	0
Sent Device Commands:	0	Resynchronisations:	0
Received Device Commands:	0		
Device Command Errors:	0		
Unknown Messages:	0		
Received Messages:	0		
Message Errors:	0		
Total Jobs:	0		
Read File Jobs:	0		
Unknown Jobs:	0		
Device Jobs:	0		
Device Read Jobs:	0		

Line Monitor

Double clicking on a COMx in the main window opens a window which visualises the data transfer to the terminals and is reserved for service purposes. This line monitor is not subject to further instructions herein. With Stop and Start the monitoring can be stopped or continued e.g. if you want to inspect a specific data sequence in detail.



```
NQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ
00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<
ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<EN
>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>0
<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<E
Q>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>
0<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<E
NQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ
00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<
ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<EN
>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>0
<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<E
Q>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>
0<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<E
NQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00<ENQ>00
```

Modem Operation

This is a brief description of how HTS carries out the setting up and abortion of a modem connection.

Connection Setup

The PC activates the line DTR and the modem responds by activating the line DSR. This handshake remains active during the whole transmission. Now the init string is sent to the modem.
After the initialisation the PC sends the dial number to the modem. The appropriate AT prefix is used.
HTS starts the modem timeout. If the connection to the terminal is successful the modem reports CONNECT and HTS can start polling the terminal(s). If there is no CONNECT within the modem timeout or the modem does not respond at all the connection is aborted and an appropriate error message supplied.

Connection Termination

After the data have been exchanged with the terminals the connection must be shut down properly. To inform the terminal that there is no more data transfers HTS sends an EOT (End of Text). The terminal responds by hanging up and the PC does the same. The PC deactivates DTR and the modem deactivates DSR. After 1.5 seconds three plus (+++)are sent to the modem to switch back to command mode. With a final ATH the modem hangs up. This happens concurrently on the terminal side.

Error Messages

The most common messages:

error 10 opening COMx !
...
Terminal y disabled
COMx not accessible!

> the COM port in question is used and locked by another program under Windows. HTS cannot cannot handle request for counters connected to this COM port.
* Terminate the other program or use a free COM port.

Multiple definition for terminal xx
Terminal disabled!

> You are trying to use a duplicate physical counter address for counter with the logical terminal number xx. HTS will not serve this counter.
* Stop HTS; Setup->Terminal Configuration: change the physical address of the appropriate counter and store.

RTC not defined for NULL
Interface disabled!
...
Terminal 1 disabled
xx not accessible!
... for all counters

> You have tried to assign more than one counters to the same COM port.
* Stop HTS; Setup->Terminal Configuration: delete all counters except one.