RTU32 User Guide

V. 1.21/ November 2006 / Doc 40209

Brodersen Controls A/S ● Industrivej 3 ● DK-4000 Roskilde ● Tel: +45 46 74 00 00 ● Fax: +45 46 75 73 36
E-mail: bc@brodersencontrols.com ● Internet: www.brodersencontrols.com

Contents

Contents **2**

1. Customer Information **5**

2. Introduction **9**

Powerful RTU and Industrial Controller with stand-alone functionality or integrated configurations 9

Open Platform with WinCE Operating System 9

3. Getting Started **10**

Configuration and default settings 10

3.1 How to connect and prepare your PC to configure the RTU32 ..... **10**

Wiring 10

Setting up your PC Ethernet driver (in WinXP) 10

First time configuration 12

3.2 Configuration parameters in the web pages **14**

Settings overview 14

Network settings 15

SNMP settings 16

MIB II Settings 16

Communication and Trap receivers 17

Permitted Managers 17

Time Settings 18

Change password 19

Edit Config File 20

Boot 22

LAN Network System Design recommendations 22

4 Getting Started with STRATON WorkBench **23**

4.1 Introduction 23

4.2 Installing STRATON WorkBench 24

4.3 Installing WIBU Software and Enter Your License Key 28

License Key 28

4.4 Install the RTU32 Hardware Library 30

4.5 Connect to the RTU32 31

4.6 Project Files / Locations 31
4.7 Application program specifications 31
5. Specific RTU32 Functions in STRATON 32
5.1 General 32
5.2 STRATON I/O Drivers for RTU32 32
5.3 COM Port Settings 38
5.4 Modbus Drivers 40
5.5 EN/IEC60870-5-101/104 Drives 40
5.6 Realtime / Realtime Clock 41
5.7 Modem Dial In/Out 42
5.8 Data Logging 46
5.9 Reading Text Files from STRATON 46
5.10 Event Binding Protocol / Redundancy Dual Binding Communication 48
General 48
Network settings in RTU32 (dual binding) 48
Procedure for setting up binding/dual binding in STRATON 49
5.11 Optional SNMP Extension Agent Driver 50
6. RTU32 cycle / scan time 51
6.1 General 51
6.2 RTU32 Scan cycle mechanisms 52
6.3 STRATON scan time 53
LocalBus scan time 53
Internal I/OBus scan time 54
7. STRATON HMI - Getting started 56
Before you start 56
HMI Procedure 56
8. RTU32 Utilities and System Event Log 61
8.1 General 61
8.2 RTU32TOOL 61
8.3 System Log/Event Viewer 62
9. RTU32 Technical Description 63
9.1 General 63

9.2 Hardware block diagrams 63
9.3 Software block diagrams 64
9.4 STRATON SoftPLC 64
9.5 RTU32 LocalBus for I/O Expansion 65
General 65
LB driver 65
LB driver event information 65
9.6 RTU32 File System 67
General 67
What does the folders contain? 68
9.7 Software Versions 71
Appendix 1 73
RTU32 STRATON applications measurement / benchmark 73
Example 1, STRATON application program without any additional serial communication 73
Example 2; STRATON application program with Modbus communication 75
ModbusRTU Slave 75
ModbusTCP Client 75
Example 3; STRATON application program with additional STRATON binding protocol 76
Example 4; STRATON application program with EN/IEC60870-5-101 communication 78
Comments to the example and measurements 79
Appendix 2 80
RTU32 I/O addressing example 1 80
Appendix 3 81
RTU32 I/O addressing example 2 81

1. Customer Information

Copyright Notice

Trademark Acknowledgement

Brand and product names are trademarks or registered trademarks of their respective owners.

Disclaimer

Brodersen Controls A/S reserves the right to make changes, without notice, to any product, including circuits and/or software described or contained in this manual in order to improve design and/or performance. Brodersen Controls A/S assumes no responsibility or liability for the use of the described product(s), conveys no license or title under any patent, copyright, or mask work rights to these products, and makes no representations or warranties that these products are free from patent, copyright, or mask work right infringement, unless otherwise specified. Applications that are described in this manual are for illustration purposes only.

Brodersen Controls A/S makes no representation or warranty that such application will be suitable for the specified use without further testing or modification.

Life Support Policy

Brodersen Controls A/S's PRODUCTS ARE NOT FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE PRIOR WRITTEN APPROVAL OF Brodersen Controls A/S.

As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into body, or (b) support or sustain life and whose failure to perform, when properly used in accordance with instructions for use provided in the labelling, can be reasonably expected to result in significant injury to the user.

2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

Brodersen Controls Customer Services

Each and every Brodersen product is built to ensure reliable performance in the harsh and demanding conditions typical of industrial environments. Whether your new Brodersen device is destined for the laboratory or the factory floor, you can be assured that your product will provide the reliability and ease of operation for which the name Brodersen has come to be known.

Your satisfaction is our primary concern. Here is a guide to Brodersen customer services. To ensure you get the full benefit of our services, please follow the instructions below carefully.

Technical Support

We want you to get the maximum performance from your products. So if you run into technical difficulties, we are here to help. For the most frequently asked questions, you can easily find answers in the product documentation. These answers are normally a lot more detailed than the ones we can give over the phone. So please consult this manual first.

To receive the latest version of the user manual, please visit our Web site at: http://www.brodersencontrols.com, choose the product in question under product search and under each product you will find accompanying data sheets, manuals, user guides etc.

If you still cannot find the answer, gather all the information or questions that apply to your problem, and with the product close at hand, call your dealer. Our distributors are well trained and ready to give you the support you need to get the most from your Brodersen products. In fact, most problems reported are minor and are able to be easily solved over the phone.

In addition, technical support is available from Brodersen engineers every business day. We are always ready to give advice on application requirements or specific information on the installation and operation of any of our products. Please do not hesitate to call or e-mail us in our offices in Denmark, United Kingdom or Germany.

Denmark:

Brodersen Controls A/S

Industrivej 3

DK-4000 Roskilde

Tel.: +45 46 74 00 00

Fax: +45 46 75 73 36

bc@brodersencontrols.com

www.brodersencontrols.com

Germany:

Brodersen Automation GmbH

Düsseldorfer Str. 138

D-45481 Mülheim a. d. Ruhr

Tel.: +49 (208) 46954-0

Fax: +49 (208) 46954-50

ba@brodersen.de

www.brodersen.de

United Kingdom:

Brodersen Control Systems Ltd.

Canbury Business Park, Unit 11

Elm Crescent, Kingston upon Thames

Surrey KT2 6HJ

Tel.: +44 (0) 20 8546 4283

Fax: +44 (0) 20 8547 3628

bcs@brodersen.co.uk

www.brodersen.co.uk

Product Warranty

Brodersen Controls warrants to you, the original purchaser, that each of its products will be free from defects in materials and workmanship for two years from the date of purchase.

This warranty does not apply to any products which have been repaired or altered by persons other than repair personnel authorized by Brodersen, or which have been subject to misuse, abuse, accident or improper installation. Brodersen assumes no liability under the terms of this warranty as a consequence of such events. Because of Brodersen's high quality control standards and rigorous testing, most of our customers never need to use our repair service. If a Brodersen product is defective, it will be repaired or replaced at no charge during the warranty period. For out-of-warranty repairs, you will be billed according to the cost of replacement materials, service time, and freight. Please consult your distributor for more details. If you think you have a defective product, follow these steps:

1. Collect all the information about the problem encountered. (For example, Product type and s/n, hardware and software version etc.) Note anything abnormal and describe the error in a product failure report.

2. Call your distributor and describe the problem. Please have your manual, product, and any helpful information readily available.

3. If your product is diagnosed as defective, make arrangement with your distributor about this.

4. Carefully pack the defective product, a complete failure report and a photocopy of proof of purchase date (such as your sales receipt) in a shippable container. A product returned without proof of the purchase date is not eligible for warranty service.

5. Ship it to your distributor.

2. Introduction

Powerful RTU and Industrial Controller with stand-alone functionality or integrated configurations

The Brodersen RTU32 controller series based on a 32-bit platform is a powerful RTU with leading edge functionality. As well as being a powerful RTU with flexible I/O designed to perform embedded data processing, control, data logging and monitoring, it is also a networking communicator for collecting, managing and communicating data via protocols on different physical interfaces upwards and downwards in an industrial environment.

Open Platform with WinCE Operating System

The new platform is based on a fan less industrial PC platform with WinCE 5.0 .NET operating system. Anyone familiar with the Windows environment will find it easy to set up the RTU. WinCE provides an open and adjustable platform with both the power and functionality required to control advanced industrial applications.

3. Getting Started

Configuration and default settings

The LAN interfaces on the RTU32 are default at delivery set as follows;

LAN1: 192.168.0.1, Subnet: 255.255.255.0

LAN2: DHCP

You can use LAN1 for setting the basic configuration parameters for the RTU32. The basic setting includes setup of network parameters as IP addresses, Subnet masks, default gateway, time settings, general username and password for accessing the web server pages and file system with FTP etc.

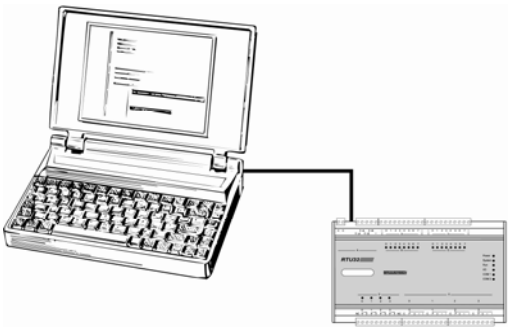
If you have chosen the optional SNMP Driver preinstalled, configuration parameters for the RTU32 SNMP Agent Driver is also included. Please see the SNMP agent driver manual for details.

The configuration is done via web pages in the RTU32 Web Server. You must use your normal web browser to setup the basic settings for the RTU32. The web pages are optimized for MS Internet Explorer browser.

3.1 How to connect and prepare your PC to configure the RTU32

Wiring

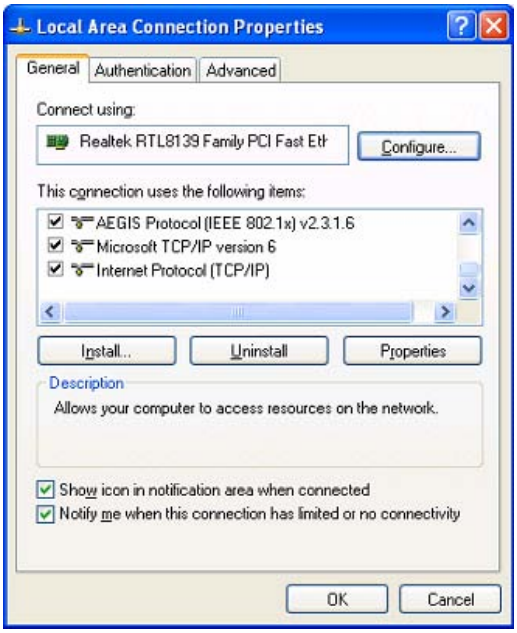
Use the included cross-wired standard patch cable to connect your PC Ethernet interface to LAN1 Ethernet interface of the RTU32. Apply power to the RTU32 – make sure that you connect the correct power supply voltage to the RTU32. If you use have a 24-48VDC version, your power supply must be able to deliver at least minimum 2A if you have a standalone RTU32 configuration.



Setting up your PC Ethernet driver (in WinXP)

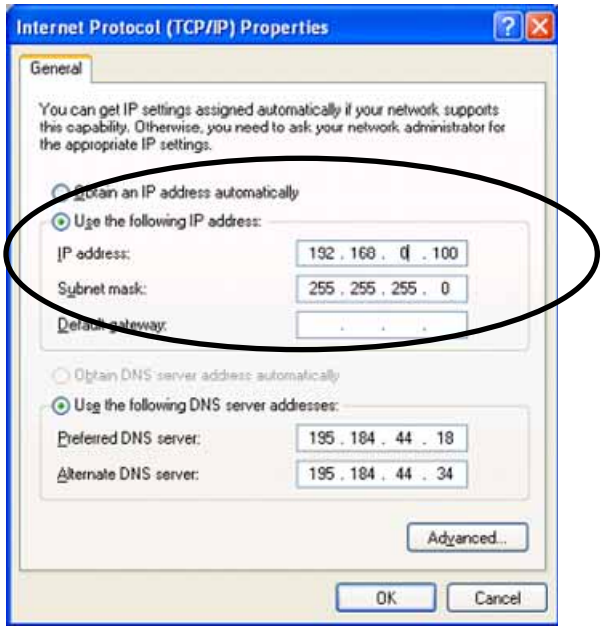
Your PC Ethernet driver must be setup to work in the same network segment as the LAN interface on the RTU32 you use. You must setup your PC Ethernet TCP/IP driver to work in the segment IP 192.168.0.x.

To setup the Ethernet driver select *start – Settings – Network Connections – Local Area Connections* and the following window will appear;



Find the Internet Protocol (TCP/IP), mark it with one mouse click and select *Properties*. A new window for setting the PC IP address will appear. Set an IP address in the same segment as the selected LAN IP address. E.g. as IP 192.168.0.100 as shown here below.

NOTE: If you want to re-configure your Ethernet TCP/IP driver back to initial settings, it is recommended that you note the settings before you change them for setting up the RTU32.

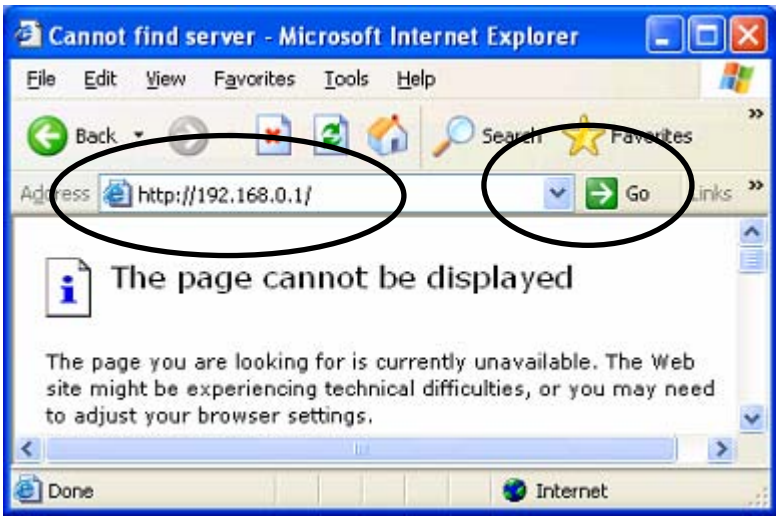


Now select *OK* and *OK* again at the Local Area Connection Properties window. Now your PC Ethernet driver is setup for communicating with the RTU32.

Now apply power to the RTU32. Wait for the RTU32 to boot up – takes approx. 45s. As a minimum the green Power and I/O LED on the front must be lit.

First time configuration

Start your Internet browser and enter the IP address of the RTU32 in the address field;



Now select -> *Go* and the first web pages will appear in your browser. Your are prompt for entering username and password – which you enter. Please store the entered username and password in a safe place for future use.

Select NEXT to continue.

Enter the username and password. Select Finish to save the settings and you will get asked to login to the RTU32 web server as shown below.

This is the login you will meet every time you try to login in the future.

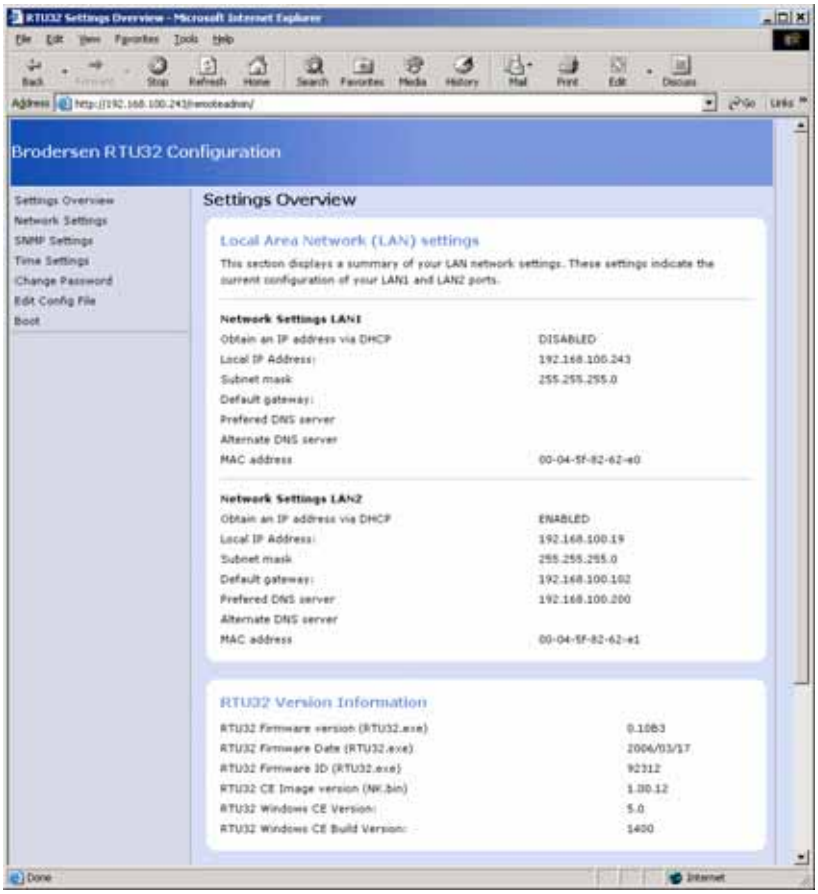
After login in you will get the first configuration web page.

3.2 Configuration parameters in the web pages

After the first time setting up the username and password, the main configuration page of the RTU32 will appear in your browser. Next time connect to the web pages you will be asked for entering username and password authentication to get access.

Settings overview

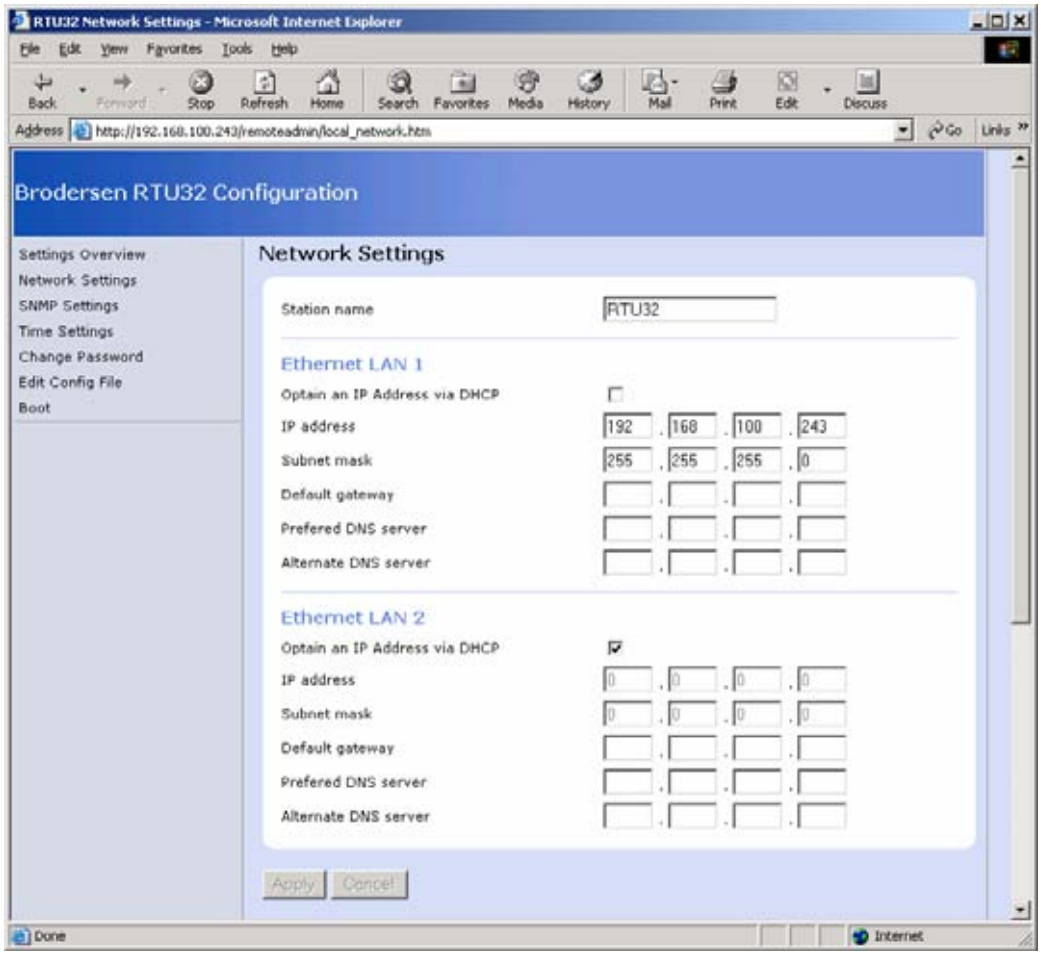
The first page that you access when you enter the RTU32 Web pages is the Settings overview page.



On this page you will get an overview of the network settings and version of the primary system files. If you are contacting your support office or distributor, you will properly always be asked for the software versions numbers.

IMPORTANT NOTE: The local IP will report 0.0.0.0 if the LAN port is not active (not connected/no connection). Check the network settings page for getting the last saved network settings.

Network settings



On the network setting page you can change the LAN1 and LAN2 settings to fit your local network. You must assign fixed IP addresses to gain access to the RTU32 with your browser, FTP client or STRATON Workbench in your LAN network. If you do not know about the IP addresses and Subnet masks in your local network, please contact your local network manager or IT department for help.

After entering the new network settings, select Apply to save the settings. Note that the new settings will NOT be activated before you reboot the RTU32. Use the reboot function on the menu at the left side of the page.

The RTU32 unit can also be assigned a unique device name (Station name), which is entered in top of the Network Settings page.

SNMP settings

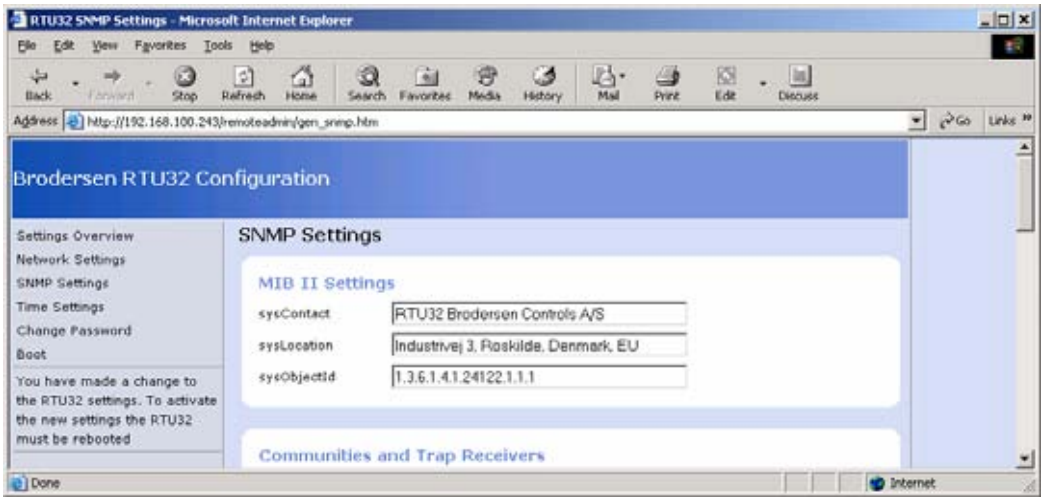
The SNMP setting page covers the basic settings for the SNMP Agent driver in the RTU32. It covers both settings for the basic WinCE Agent Driver which report network parameters and the RTU32 SNMP Alarm Extension Agent. The basic RTU32 SNMP Agent is handled be the WinCE OS and do include the standard functions for network data and statistics implemented by Microsoft©. The SNMP Alarm extension agent is used for application specific Trap alarming, Get/GetNext and Set functions handled in the STRATON SoftPLC program.

The general WinCE SNMP Agent is always enabled in all RTU32 types. The STRATON SNMP Extension Agent used for sending Traps etc. defined in a STRATON application program is ONLY supported is you have bought a RTU32 with this option enabled.

The configuration parameters for the SNMP Agent Driver are;

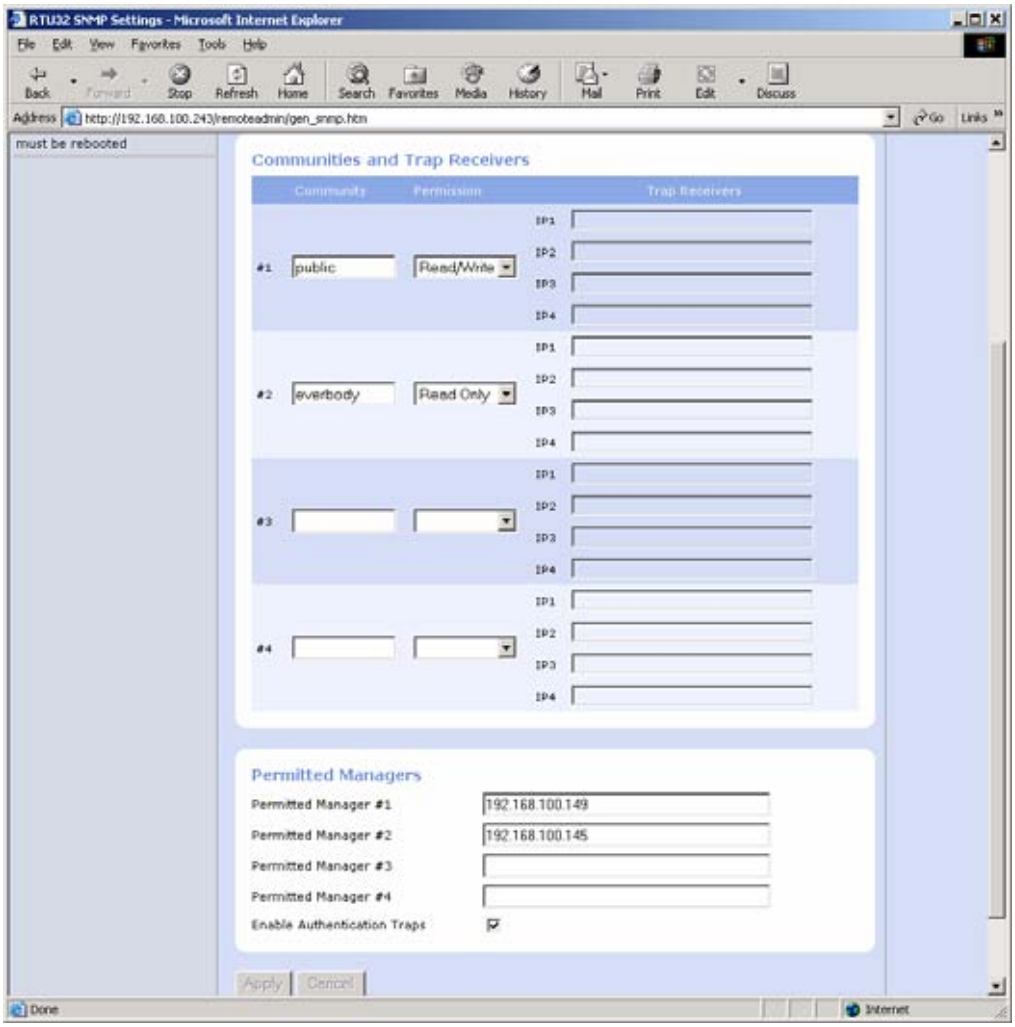
MIB II Settings

MIB II is the basic WinCE SNMP Agent. You can define the some information in the basic agent. The informations are Contact, Location and ObjectId. NOTE: Do not remove the object id unless you have full control over the SNMP Manager settings



Communication and Trap receivers

In this configuration area you are able to setup up to 4 communities with each 4 SNMP Managers IP addresses. Note that the community "public" is the default one and normally know by any SNMP Management software. In addition each community can be adjusted to different levels of access permissions.



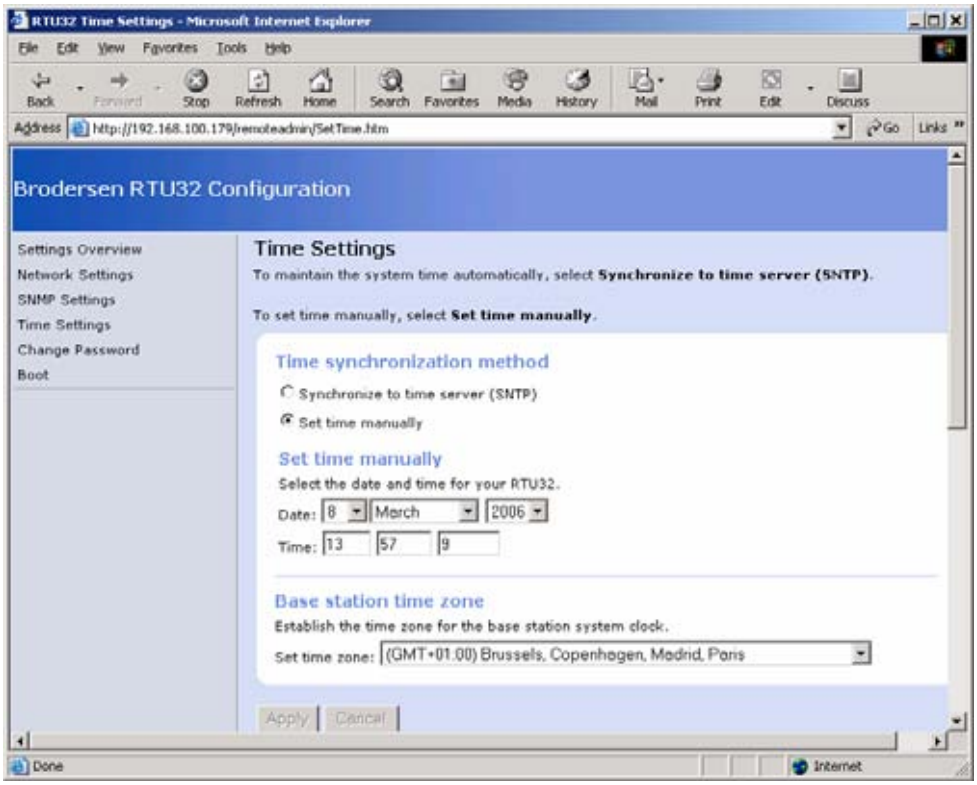
Permitted Managers

You can control access to Get/GetNext request and Set commands by adding Permitted Managers IP addresses. If no SNMP Managers are entered all Managers have access.

If you enable the Authentication Trap function, the permitted managers will get a Trap report if unauthorised has been attempted.

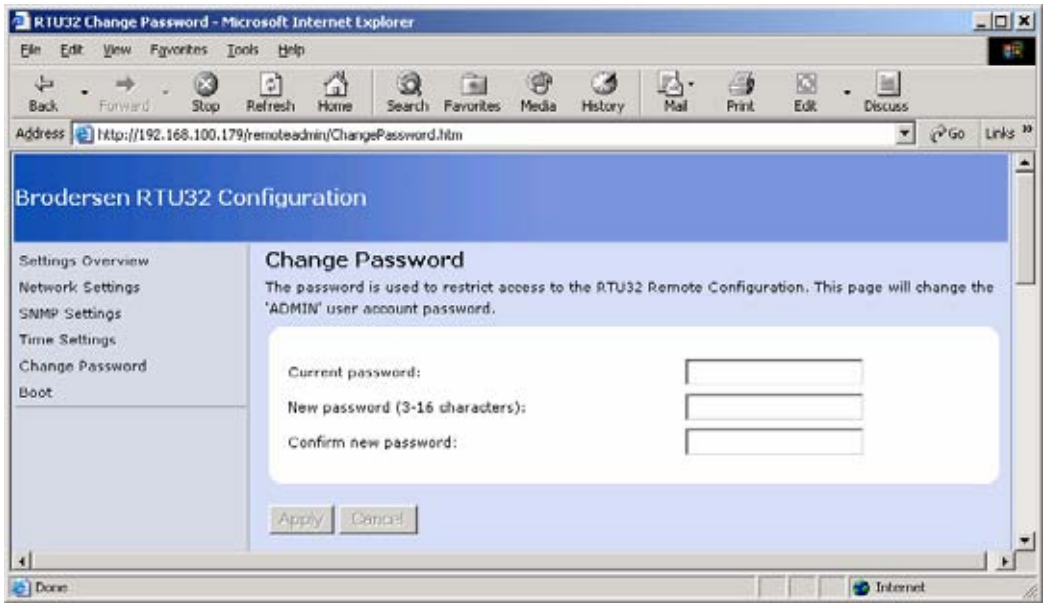
Time Settings

The time settings web pages is used for adjusting the RTU32 realtime clock. You can choose to enter then time manually or set the time from a network time server using SNTP. If you use the SNTP you have to enter the domain name (like www.example.com) or IP address of the time server.



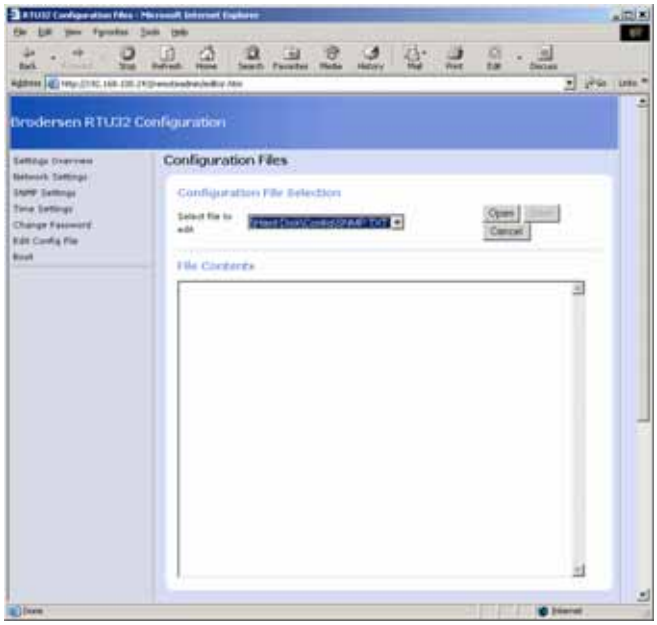
Change password

You can change the username and password for web server access, FTP etc.

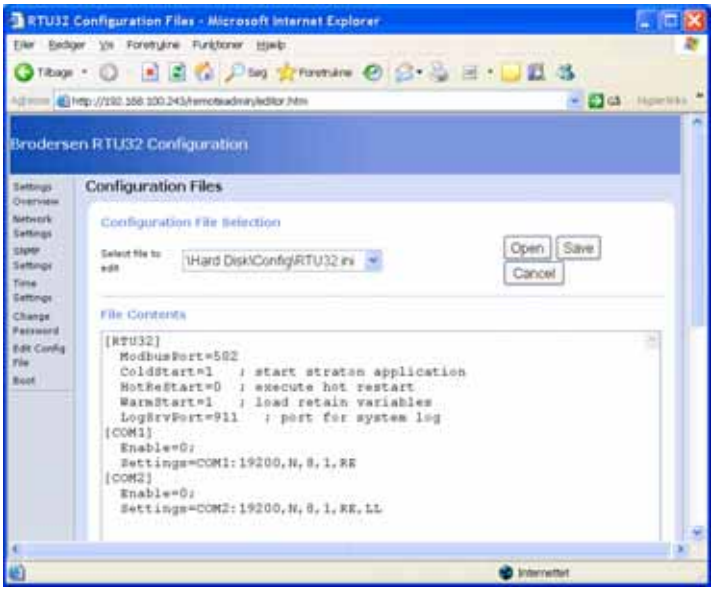


Edit Config File

The Edit Config file web page is used for make changes and adjustments in RTU32 configuration files.



For basic configuration parameters in the RTU32, the RTU32.ini file is opened.



The settings in the RTU32.ini are significant as they include some basic settings for system log port, use of ModbusRTU Slave driver in STRATON etc.

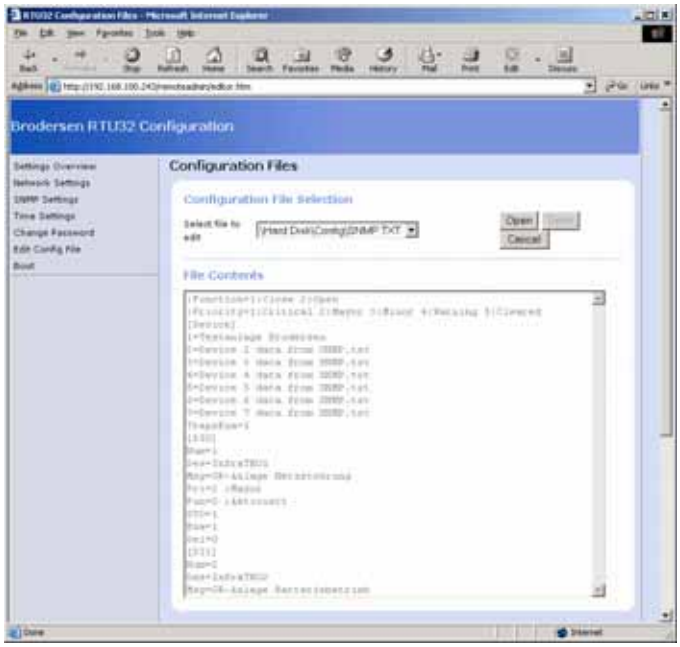
The settings are;

```
[RTU32]
  ModbusPort=502
  ColdStart=1          ; start STRATON application at boot
  HotReStart=0         ; execute hot restart
  WarmStart=1          ; load retain variables
  LogSrvPort=911       ; port for system log
  LockIOconfig=0       ; '1' Freeze LocalBus configuration
[COM1]
  Enable=0;
  Settings=COM1:19200,N,8,1,RE
[COM2]
  Enable=0;
  Settings=COM2:19200,N,8,1,RE,LL
```

COM1 or/and COM2 must be enabled to run ModbusRTU Slave from STRATON application. The COM ports are in this case locked to Modbus and cannot be used by any other serial driver in STRATON. In order to use COM port for communication to the WorkBench, you also must enable the COM port as it uses also the Modbus protocol.

The settings for COM port are the same as in COM port setup in STRATON – see COM Port settings section for details.

For SNMP a special configuration file SNMP.txt can be used. To edit the file you must select and open it.



After opening the file, you can edit it like in any other text editor. Select save to save the configuration file on the RTU32.

The format of the default SNMP.txt file is described in the SNMP Demo description. Please note that in the demo application, you do not need to reboot the RTU32 after changing settings in the file – the application uses a function which online use new settings.

Boot

After changing parameters on the configuration pages, you must unless otherwise is defined, run the reboot function to activate the changes.

After booting the RTU32 you can start working with STRATON Workbench to create you application programs.

LAN Network System Design recommendations

After being familiar with the basic settings of the RTU32, you are ready to make a system of 2 or more RTU32 on a LAN of WAN network. Before you start, it is recommended to make a system design plan. Beside your specific application system requirements you should make a table of the RTU32s in your network with RTU32 unit names, physical locations, IP address etc.

NOTE: The SNMP.txt file is just an example of how you in your STRATON application can use files for reading/writing variables.

4 Getting Started with STRATON WorkBench

4.1 Introduction

The STRATON programming tool fully supports EN/IEC61131 and is used for making SoftPLC programs in the RTU32. The application program kernel is implemented and runs in WinCE real-time task. STRATON offers complete SoftPLC functionality and supports all features needed in today's industrial environment. STRATON supports programming languages such as Structured Text, Function Block, Ladder, Instruction List and Sequential Function Chart. The STRATON SoftPLC supports cold restart, hot restart and on-line changes.

The STRATON Workbench is delivered in 2 ways:

1. Stand-alone STRATON Workbench. Are supplied and supported by us.
2. STRATON Workbench as an integrated part of ZenOn SCADA editor. Are supported by your ZenOn distributor.

Note that all examples and guidances refer to the stand-alone STRATON Workbench. The ZenOn integrated STRATON Workbench however fully compatible with RTU32 – is the same software package. Only details as file placements etc. is different. The STRATON functions in general is exactly the same.

STRATON support also general Windows PC and Server environment. For the RTU32 target a range of special and RTU dedicated functions are added.

The STRATON Workbench is used for configuration protocols, programming and debugging. It supports several tools for multi-program handling and documentation. It is also a powerful tool for complete system design and programming, providing unique functions for event based communication. The Global Binding Editor makes it possible to publish and subscribe variables in a large network with minimum communication load. The events are time stamped and can also be used directly in ZenOn SCADA HMI applications.

Programming, debugging and upload and download of application programs can be done remotely via Ethernet or RS232.

To make it possible to download and execute runtime applications in the RTU32, a runtime virtual machine is implemented in the RTU32. The RTU32 target specific functions used in the programmer environment (called the STRATON WorkBench), some hardware specific definition files are installed.

The STRATON Workbench is a licensed software tool which requires a dongle on your workstation. You can choose the different licenses which are reaching from low amount of I/Os to unlimited I/Os. The I/Os are calculated as 1:1 number of variables (a Boolean and a WORD equals 1 I/O). In addition any declared driver parameter is counted as one I/O (i.e. each ModbusRTU register is 1 I/O).

The RTU32 STRATON runtime license is included in the RTU32 price – once you have bought a STRATON Workbench you can program as many RTU32 as you like. Each application program can only include the I/O counts within your WorkBench license

limitation. Be careful about the use of I/Os – you will even in smaller application quickly count up I/Os.

The STRATON workbench include in addition some standard features like Modbus suite of drivers, EN/IEC60870-5-10x suite of basic link drivers, debugging facilities, project documentation generator and much much more. In addition some advance feature packages are available. Please see the STRATON documentation for details.

4.2 Installing STRATON WorkBench

Insert the installation CD in your drive.

If the installation program does not start automatically, press ”START” select ”This computer”. Double click on the STRATON icon on your CD/DVD drive as shown on fig. 1

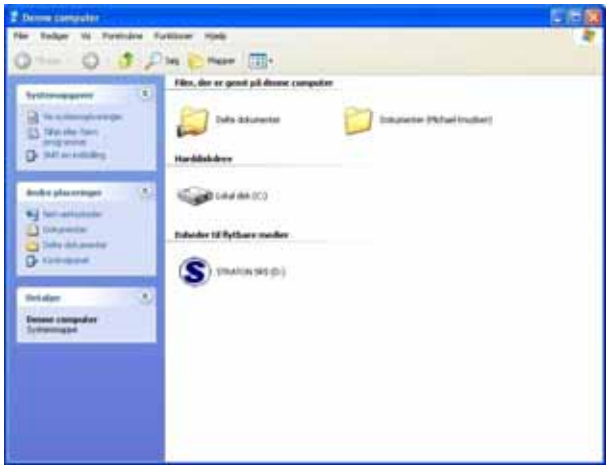


Fig. 1

In the installation window click on ”Install the STRATON Workbench” and follow the instructions.

Fig. 2

Click on ”Finish”.

In the installation window click on ”Install THE WIBU driver” to install the USB hard key dongle.

Fig. 3

Follow the instructions.

If support of other languages is needed, select them when the screen below appears.

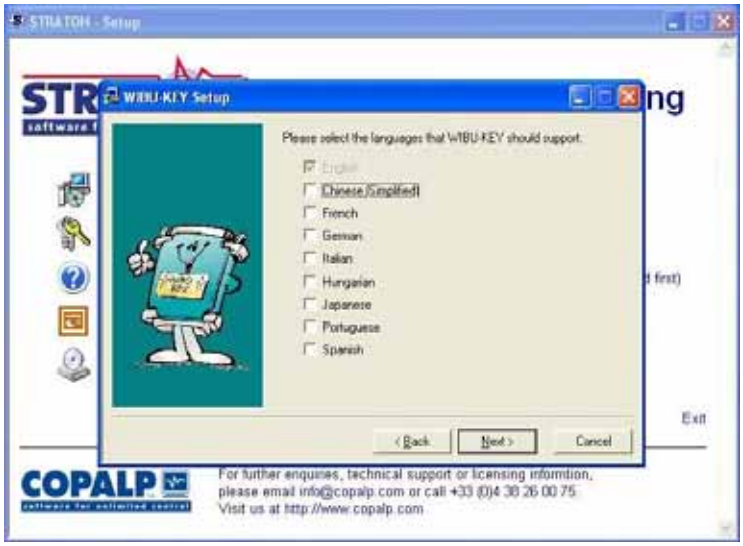


Fig. 4

Click ”Next”.

If it is necessary choose a different destination folder, and click ”Yes” to create the new folder. See fig. 4

Fig. 4

click on ”Next”

When the screen below appears, just click ”Next”. No changes are necessary.

Fig. 5

Follow the instructions on the screen. When finished the screen below should appear:

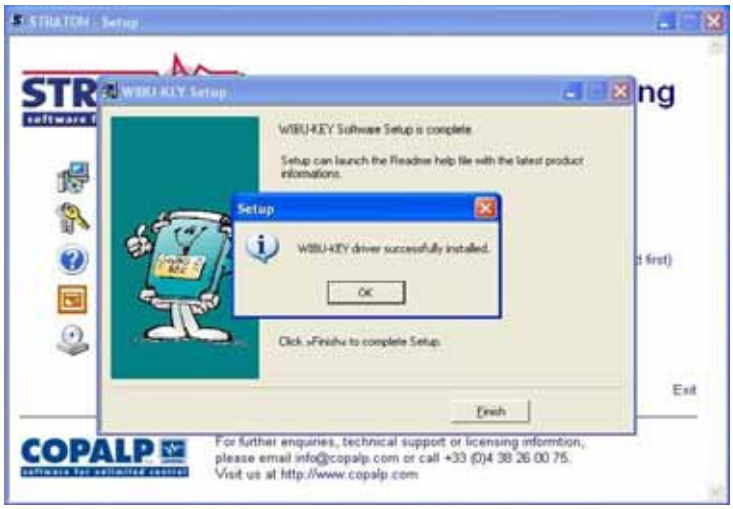


Fig. 6

The installation is completed and you can click on ”Exit”.

Fig. 7

4.3 Installing WIBU Software and Enter Your License Key

License Key

Before using STRATON a license-key have to be entered. (Included in the STRATON CD-box)

The license-key removes the demo-mode restrictions.

Choose ”START” -> ”Programs”-> ”STRATON” and click on ”License” as shown below

Fig. 8

Select ”Use a local license (for this product only)” and click on ”Change..”.

Type in the license key and click ”OK”. See fig. 9

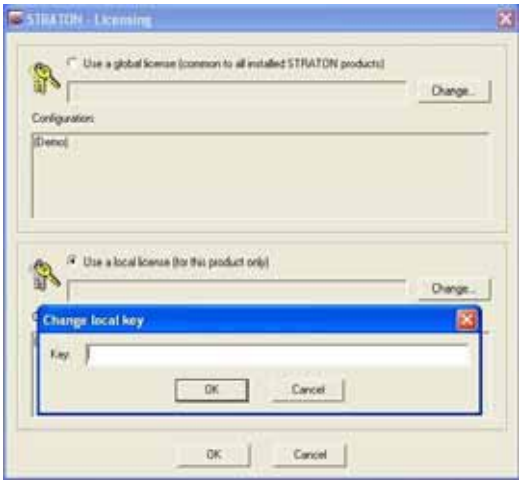


Fig. 9

NOTE: The license key is case sensitive.

Insert the dongle in the USB port. The ”New hardware found” screen should appear.

Fig. 10

Click on the box and follow the instructions on the screen.

Start STRATON by pressing ”START” -> ”Programs” -> ”STRATON” and click on ”STRATON” as shown in the screen below

Fig. 11

Select the language in which you prefer to operate the program in and press ”OK”.

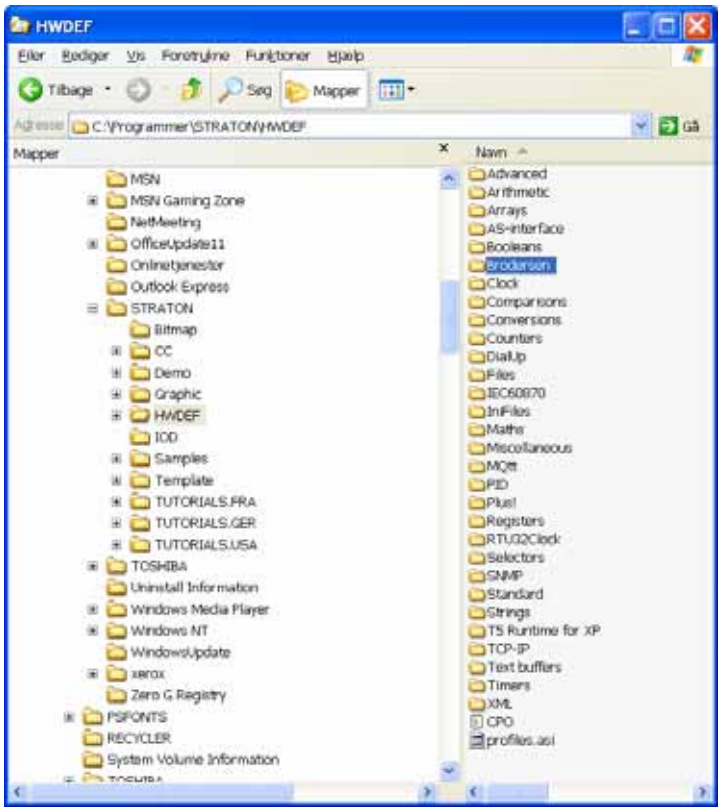
Fig. 12

4.4 Install the RTU32 Hardware Library

On the CD which is included in the RTU32 package, you will find the specific RTU32 target hardware definitions which must be installed to support the RTU32.

After installation of STRATON Workbench you must copy the files on the enclosed CD in the../HWDEF/ folder to ../Program files/STRATON/HWDEF/ on your Workstation/PC.

You can check if the files are installed correctly if the HWDEF folder contains the folder Brodersen which is one of the RTU32 specific hardware library folders.



After installation of the library files, the RTU32 specific drivers, support for I/Os etc. will be available in the STRATON WorkBench.

If you use ZenOn with integrated Workbench, you must address the ZenOn documentation for file details.

4.5 Connect to the RTU32

In order to connect STRATON WorkBench to the RTU32, you must enter the IP address you have defined for the RTU32 and the default port address 502. Please note that the programming communication uses the same port address as ModbusTCP. It can fine interact parallel with a ModbusTCP connection as the programming protocol has defined a special T5 id in the protocol part for special incremental frame counter values.

4.6 Project Files / Locations

Each STRATON Project is placed in a Folder with a name that you assign. The demo projects are placed in the ./Program Files/STRATON/Samples/ folder but you can choose to place them according to your requirements.

On the CD you will find some RTU32 Demo projects showing different type of simple applications using different function like Modbus Drivers, Brodersen I/O etc. If you want to use them you must copy them to the Workstation/PC before opening them. Some demo files include also documentation.

4.7 Application program specifications

The STRATON application programs have some general specifications which are defined below;

Max size of program

The maximum size of an application program is limited to 64kb of compile code. The limitation is not restricted to actual lines of code as the lines do not equal the size of the executable program downloaded to the RTU32.

Max number of programs

The max number of program and sub programs in a project is limited to 32767.

I/O limitations

If you have a STRATON WorkBench license with I/O limitations of e.g. 512 I/O – this is the limitation. If you are using an unlimited license the max number of I/Os are 65535.

Max Binding/Dual binding connections

For the number of Binding connections allowed to connect to the RTU32 is default set some limitations in order control the use of RAM memory. The RTU32 is limited to max 32 continuous sockets. The limitation covers all TCP socket and does also include any other TCP socket used by other TCP/IP connections including the WorkBench connection.

5. Specific RTU32 Functions in STRATON

5.1 General

In general it is highly recommended that you take a training course in STRATON. That will save a lot of time and is a good investment because you can get familiar and right away take advantage of all the STRATON powerful features.

Contact your local distributor for STRATON training exercises.

Beside the knowledge of STRATON programmer environment and EN/IEC61131-3 programming languages, there are some functions and features which is RTU32 specific.

5.2 STRATON I/O Drivers for RTU32

All I/Os including status are stored in an independent database in the RTU32. The database reflect the settings of the physical I/Os. The Internal I/OBus and the LocalBus for expansion modules do update the I/Os. The STRATON I/O driver handles the access to this database. You can program your total application without any connected hardware. With this in mind you must be sure that you have the right I/O configuration on your RTU32 hardware when you try to run and debug your application program on a RTU32.

The database can optionally be access by other applications. A specific programmers API called WTOOL is then used.

In order to control and differentiate new updates and versions of the I/O driver in STRATON, the last digit in the names are used as version. You can keep older versions in STRATON by renaming the old hardware definitions before installing new ones (e.g. Rename "Brodersen" to "Brodersen_v0").

I/O database access

The STRATON I/O driver for RTU32 is implemented to support two ways of accessing the I/O database in the RTU32. The ways are called I/O Board and I/O Profiles.

I/O Board is used for declaring I/O variables after the physical I/O layout. You are able to add e.g. a 16DIO expansion module from a predefined board list.

I/O Profiles gives the possibility to declare a single input or output and include additional scaling possibilities for analogue I/Os. Profiles are also used when you want to read status information from the RTU32 and read the DIL switches.

NOTE: Inputs can be declared by both Board and Profiles, but output must only be declared by one of them!

I/O database layout

Before you start programming it is important to understand the I/O database layout and how the addressing is structured. The database has basically 10 data types;

ST: Status word for I/O and read DIL switch settings

DI: Digital inputs

AI: Analogue inputs

YI Virtual inputs

ZI Auxiliary inputs (include also CI – 32bit counter inputs)

DO: Digital outputs

AO: Analogue outputs

YO: Virtual outputs

ZO: Auxiliary outputs (include also virtual outputs for resetting counters)

VIO: Virtual I/O used for database access

The virtual input and output register YI, YO and VIO are normally not used (reserved for data access to special applications accessing the basic database API).

Auxiliary registers are used for special data types. Counters are read and reset through ZI and ZO registers.

All data types are called **Modules** and are defined in **Word values (DINT)**. Each data type is addressed individually starting with 0 (zero). Each I/O Board has its own module layout which is defined in the table below.

I/O Boards	DI	AI	DO	AO	ZI	ZO	UCL-type
RTU32_04AO				4			UCL-04AO
RTU32_08AI		8					UCL-08AI
RTU32_08AIxP		8					UCL-08AIxP
RTU32_08DI	1						UCL-08DI.Ax
RTU32_08DO			1				UCL-08DO.Rx
RTU32_8DIO	1		1				UCL-8DIO.Px
RTU32_16CIS	1				2X16	1 (RC)	UCL-16CIS.P
RTU32_16DI	1						UCL-16DI.Dx
RTU32_16DIO	1		1				UCL-16DIO.Px
RTU32_16DO			1				UCL-16DO.Rx
RTU32_26IO2	1	4	1	2	2X16	1 (RC)	UCN-26IOA
RTU32_32DI	2						UCL-32DI.Dx
RTU32_32DO			2				UCL-32DO.xx
RTU32_36IO	2		1				UCL-36IO.Px

Addressing example

To give a practical example, 2 different RTU32 I/O configurations with module addressing are shown in Appendix 2 and 3.

Layout for the status register (ST)

In the I/O database the status word (ST) reports relevant status values:

Module 0: I/O bus status;

Word value = 0 OK, no errors

bit 0	Unknown LB module type, can't build database.
bit 1	LB scan error, Typical one or more LB modules are missing after LB start.
bit 2	STRATON accessed database module number is not present.
bit 3	IO configuration mismatch
bit 4..15	Reserved

Module 1: Reserved.

Module 2: RTU32 DIL switch settings (switch 0 = bit 0, etc.).

NOTE: The DIL switches located on the downwards side of the RTU32 are read in the status register (ST) module 2 (third word).

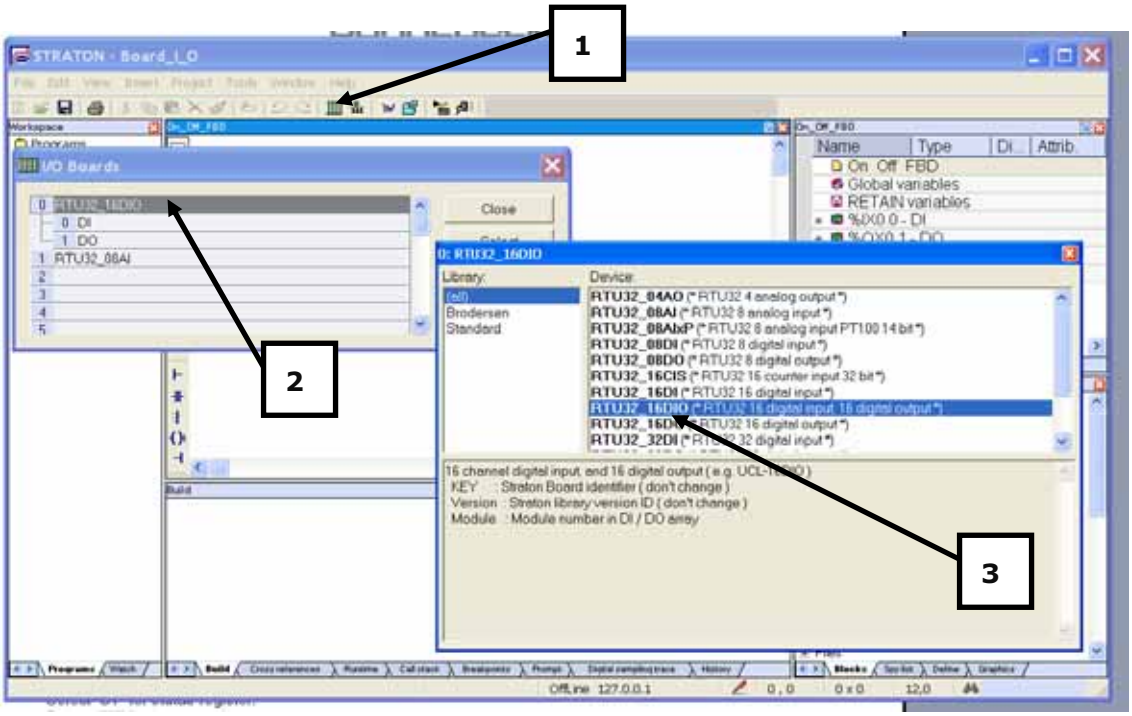
IMPORTANT NOTE: The 2 first DIL switches are assigned for STRATON application program run mode settings. Here will be implemented features for start and stop of application program including cold and hot restart features.

STRATON Board I/O Driver

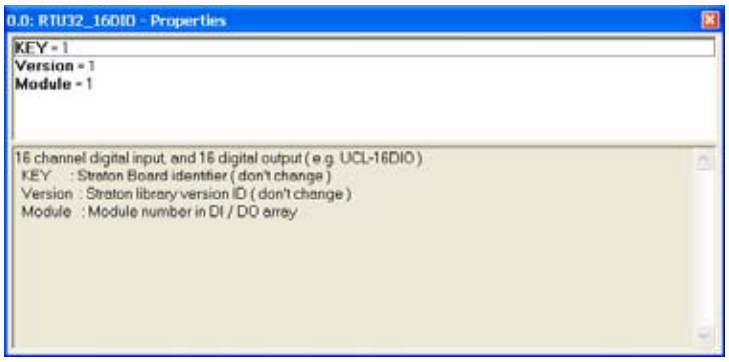
The board I/O driver makes it possible to declare I/Os in sections as the physical board layouts. I.e. is it possible to add all 32 digital inputs in one process using the I/O Boards driver. At declaration it automatically declares the variables and put them in the variable list.

Procedure for declaring board I/Os;

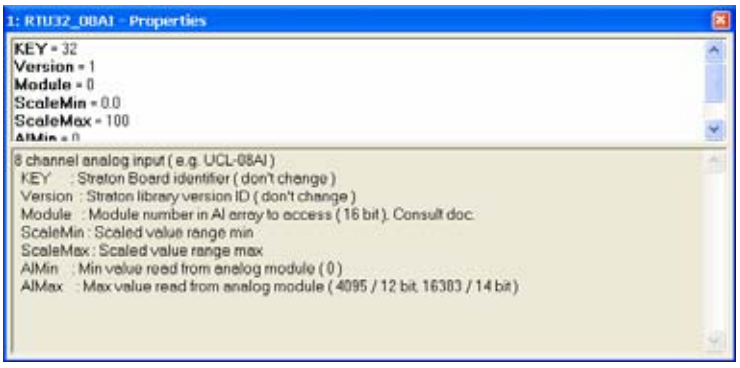
- Select Open I/Os (1)
- Double click on the section 0 to add your first board (2)
- From the list of I/O board select the one you want (3)



After the board is selected you will see it in your list. Now select properties to configure the board. If the board you have selected have different type of I/O they will be shown individually in it own line. Every data type require configuration.



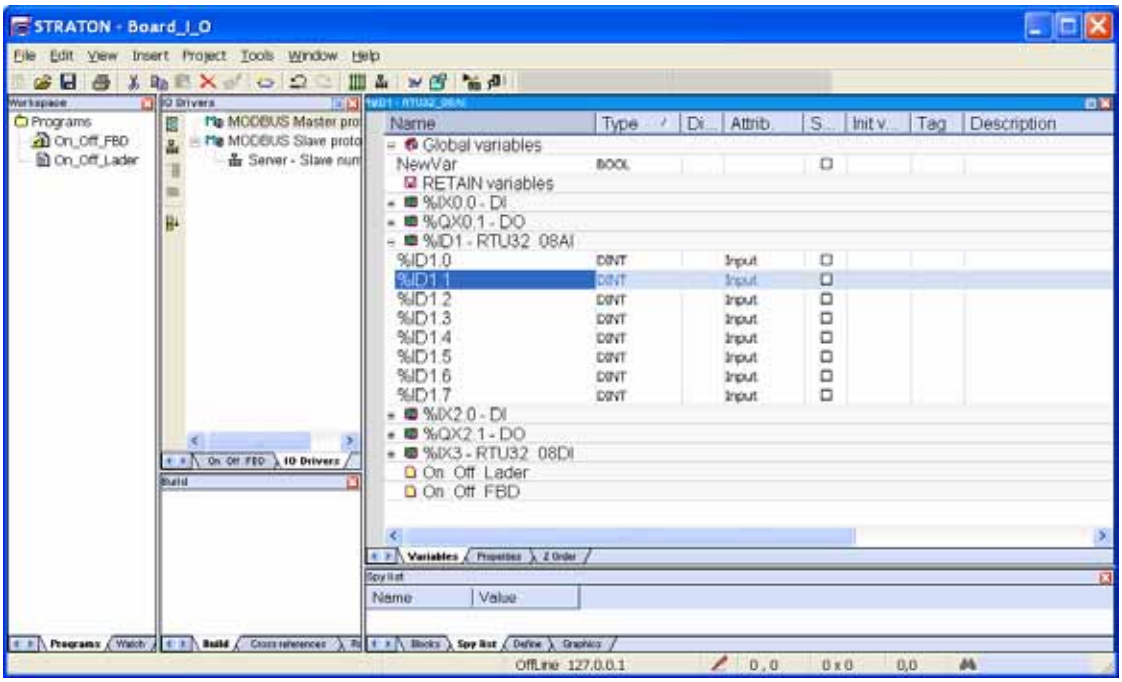
The properties for digital I/Os it quite simple. Only the Module address needs to be setup. Use the addressing example to determine the Module address in your application.



The properties for analogue I/O include more configurations. Beside the Module address of the first analogue channel, you can configure the scaling of the analogue I/Os. Please note

that the scaling will cover all I/O in the actual board section. On some board I/Os it is possible to set the range of every input and output individually. This is e.g. the case for Internal I/Os and a newer analogue expansion modules.

After setting up the boards, you will find the all the board I/Os in the variable list.



The I/Os are now ready to use in your application program.

For some of the analogue input I/Os you will find some qualifier input which is used for status information. They report e.g. underflow and overflow.

NOTE: Board variables are not compatible with ZenOn – they cannot be read/write in the ZenOn database. Only profile I/O driver has this option.

Selection of AI input type and range can only be setup in board I/O. If you are using profile I/O in your application, you can use the board I/O to setup the analogue inputs. You must then set module = -1. The I/Os will be ignored and only the AI configuration used.

STRATON Profiles I/O Driver

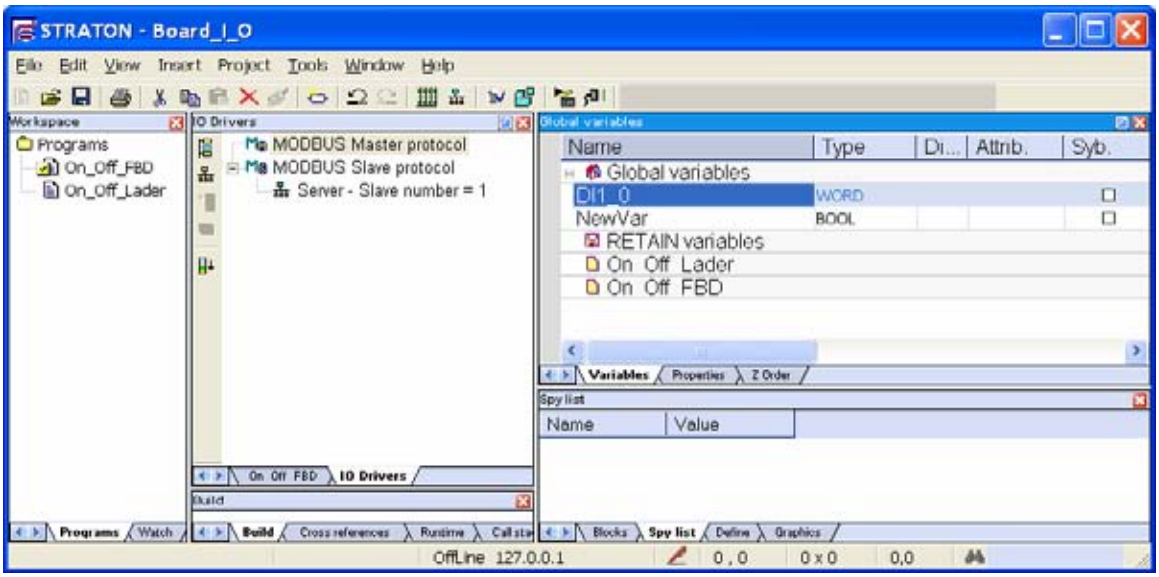
The Profile I/O driver is used for declaring I/Os one by one directly in the variable list. Any I/O digital or analogue can be declared this way.

If you want status from our database or read the DIL switch settings, you must always use the Profile driver.

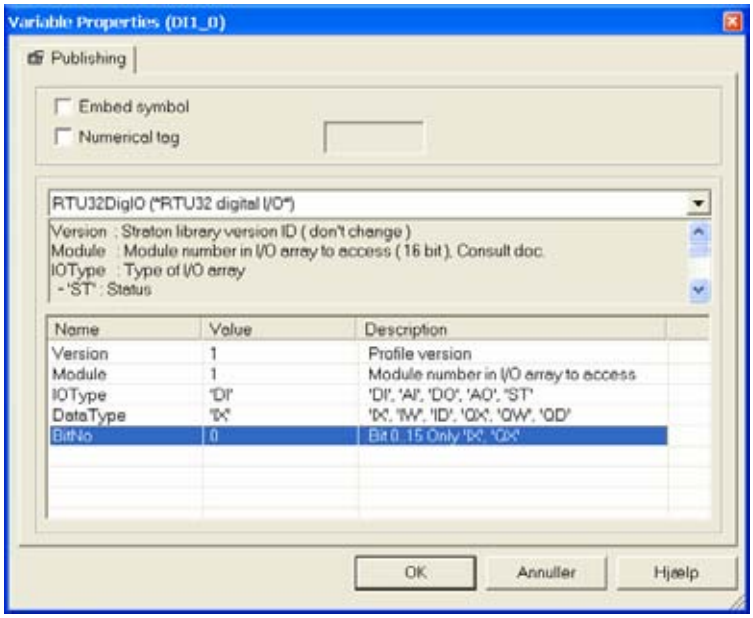
Procedure for declaring profile I/Os

In the Variable list you add new variable under the Global variable structure. It is recommended to use Global Variable structure to have access to the I/O from any program in your STRATON project.

Double click on the variable name and enter a proper name for your I/O – e.g. DI1_0 as for the first digital input bit in the second DI module. Set the variable type to WORD.



Right click on the DI1_0 variable and select Properties.



Select "RTU32DigIO" for digital I/Os, set Module = 1 (second digital input module).

Select "DI" for status register.

Select "IW" for input word.

Say OK to store the settings.

Now the physical I/O is defined as a variable and can be used in your application program. If you want a single bit value you can either declare the variable as a bit (IX) or you can just write "DI1_0.2" to read the 3rd bit value.

The same procedure is used for reading the status words and DIL switches.

Counter inputs

Some I/O boards have support for counter inputs. Auxiliary inputs (ZI) are used for reading 32bit counter inputs. The counter values are stored in registers (modules) and can be reset by using a virtual output (ZO).

Reset of counters is done on an output raising edge. Make sure that the reset output is hold for minimum 2sec in order to ensure that that one I/O scan has been performed. This could be done by using a timer in the STRATON application program.

5.3 COM Port Settings

In general the COM port setting is defined in STRATON when using the different drivers. This goes for ModbusRTU Master and the EN/IEC60870-5-101 drivers. When using these drivers the COM port parameters are defined as "COM1:9600,N,8,1". In addition some specific control and debug parameters is added for the RTU32.

The COM port setting strings follow the normal STRATON conventions. COMa:b,c,d,e (e.g. COM1:9600,N,8,1).

a) COM port number:	1..8
b) Baud rate:	110, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400,
	57600, 115200
c) Byte size:	7, 8 (bit)
d) Parity	N, E, O (none, even, odd)
e) Stop bits	1, 2

A range of additional parameters on the COM port settings is supported to control;

- Setup the COM port for modem dial support – enable the Modem driver.
- Hardware handshake signal on null modem driver.
- Provide communication log in the general RTU32 System Log.

The extensions for these functions and features are;

- When using the modem driver a 'M' is added (e.g. **M**COM1:9600,N,8,1)
- RTS / CTS hardware handshake control on the null modem serial driver is supported and enabled by the options as follows:

  'RD' RTS is kept inactive (low) at all time.

  'RE' RTS is kept active (high) at all time.

  'RT:Leading:Trailing' RTS is inactive when receiving data, and become active when transmitting data. The RTS Leading setting defines the delay from activating the RTS to the first character is transmitted. The RTS Trailing setting defines the delay from the last character is transmitted to RTS is deactivated.

  Leading and trailing time is in units of ms. - e.g. COM1:9600,N,8,1,RT:50:10

  'RC:Leading:Trailing' RTS is inactive when receiving data, and is activated when the RTU wants to transmit data. After activating RTS, the RTU will wait for CTS to become active, before start transmitting. The RTS Leading delay is still valid in this mode, and an adjustable delay from CTS is activated to first character is then possible. However by setting the Leading time to zero, there is no unnecessary delay from CTS to first character (like normal RTS / CTS function). After activating RTS the RTU wait up to 10 sec for the CTS signal. If timeout occur, transmission is discarded. The RTS Trailing setting defines the delay from the last character is transmitted to RTS is deactivated.

- Logging of serial data in the System Log.

  'LL' Log Link data. All bytes transmitted and received on the COM port, are logged in the internal log message system, and is available using telnet at port 911. This is primary for debug purpose.

  Examples: MCOM2:9600,N,8,1,LL - COM2:57600,E,8,1,RT:30:10,LL

The settings string is applicable for both build in STRATON drivers and Brodersen drivers.

Note! STRATON WorkBench has a limitation of max 31 chars in the settings string.

5.4 Modbus Drivers

The RTU32 STRATON runtime support function for 4 type of Modbus protocols;

- ModbusRTU Master
- ModbusRTU Slave
- ModbusTCP Server
- ModbusTCP Client

The drivers are setup using the Fieldbus Configurator in STRATON. Refer to STRATON training and programming examples.

ModbusRTU Master

The RTU32 support several Master drivers. You can in fact setup as many Master drivers as you have COM ports available.

ModbusRTU Slave / STRATON serial programming port

The RTU32 support maximum 2 Slave drivers. The number of Slave drivers requires setup in a configuration file in the RTU32. Default it does not to support any ModbusRTU Slave drivers. Options are;

- No ModbusRTU Slave drivers – COM 1 free for any use (default)
- 1 x ModbusRTU Slave driver fixed to COM1.
- 2 x ModbusRTU Slave driver fixed to COM1 and COM2.

The ModbusRTU Slave port is also the serial STRATON programming port. It means that if you want to use serial programming port, you must setup COM1 to be a dedicated programmer port and ModbusRTU Slave port. It can in this case not be used for anything else.

To setup the wanted ModbusRTU Slave driver you must open the RTU32.ini file in the webpage text editor. See section "Edit Config Files" for details.

5.5 EN/IEC60870-5-101/104 Drives

Utility protocol according to EN/IEC60870-5-101 and -104 are implemented as a basic link driver in the RTU32. The application layer which include all the specific handling of request and events communication is managed in the STRATON application program. A range of STRATON functions and example are provided for the different protocol functions and ASDU handling.

STRATON include drivers for;

- EN/IEC60870-5-101 Master
- EN/IEC60870-5-101 Slave

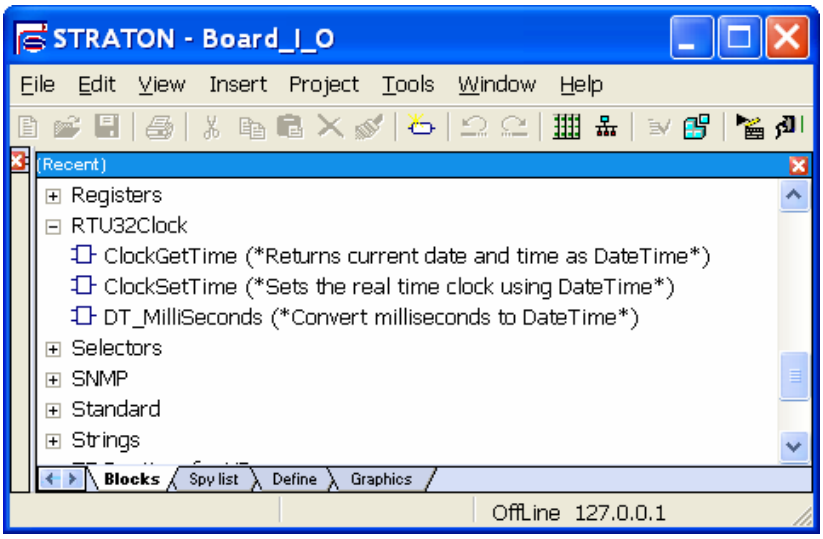
- EN/IEC60870-5-104 Server
- EN/IEC60870-5-104 Client (optional)

The basic link drivers includes handles for the link driver where you can define buffers, link address etc. The fact that the serial link is similar to the EN/IEC60870-5-103, provide the option to implement application layer support for this protocol as well.

More information and specification can be obtained in the STRATON EN/IEC60870 Driver manual. A dedicated configuration tools for setting up EN/IEC60870 drivers will be available late 2006.

5.6 Realtime / Realtime Clock

The RTU32 has a hardware real time clock and is controlled in the operating system. It provide in RTU32 support for milli second resolution in order to make accurate time stamping. From the STRATON application program you have access to the real time clock via some special functions as shown below. You can both read and set the clock.

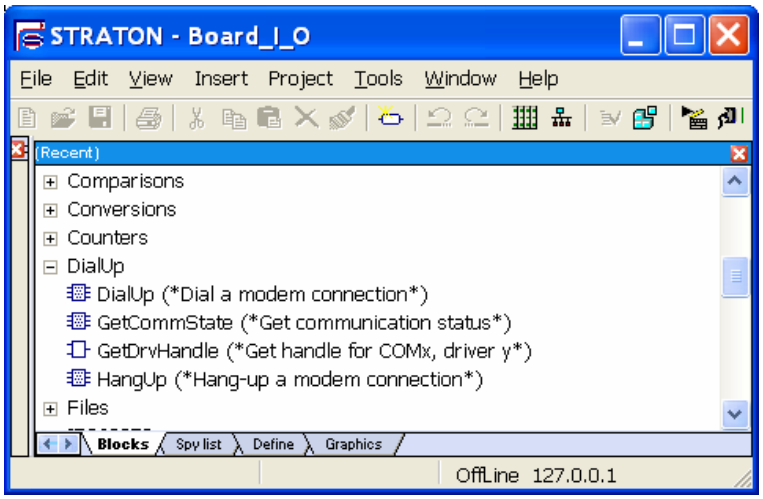


On the configuration webpages you can also define how to synchronise the clock. For now options for manual setup and SNTP Server synchronisation is possible.

5.7 Modem Dial In/Out

Dial functions for communication with serial drivers over PSTN and GSM modems are provided.

The low level modem handling (send AT commands, connect detection etc.) is done by firmware. However it is up to the STRATON programmer to decide which protocol driver should be activated, when and where to dial, and respond to modem connection state. A number of STRATON 'C' functions are available for this.



The modem function is enabled by adding M to the COM settings. See details in COM settings section.

After enabling the modem dial driver, you must add a Modem driver handle in your application program (GetDrvHandle). Your dialup function must now refer to this handle. The DialUp function include all the necessary parameters to establish and control a dial-up or dial-in session.

GetDrvHandle (Get driver handle).

This function is used to get handle of build-in protocol drivers, like ModbusRTU master or slave. This should be used when operating a build in protocol driver (e.g. dialing). To obtain the handle, COM port number and driver number must provided. Note! For now only one driver is supported and must be set to 1.

GetDrvHandle (*Get handle for COMx, driver y*)
IN COM:DINT Drv:DINT
OUT Handle:DINT

Input parameters:

Com: (DINT); COM port number from which a handle should be obtained.
Range 1..8.

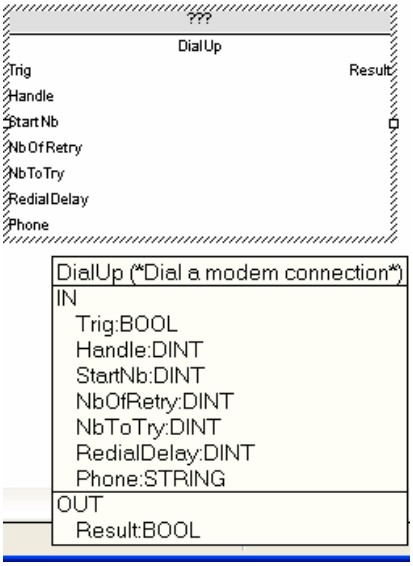
Drv: (DINT); Driver number from which a handle should be obtained.
Range 1..3 (only 1 is supported today)

Output parameters:

Handle: (DINT); Handle of the COM port driver combination.
If the result is -1 the COM port driver number combination is invalid.

Remarks: The function should only be called once at start of STRATON application.

Dialup - Start a dial-up connection for one of the protocol drivers.



Input parameters:

Trig: (BOOL); '0' to '1' transition will start dial-up

Handle: (DINT); Protocol driver to start a dial-up. Result from one of the protocol driver configuration functions.

StartNb: (DINT); Index into the phone number list to dial. First phone number is indexed 0.

NbOfRetry: (DINT); Number of times to retry the same phone number in case of missing connection. Range 0..10

NbToTry: (DINT); Total number of Phone numbers to try. If one only the main number is dialed. Range 1..10

RedialDelay: (DINT); Seconds to wait before redial. Either the same or succeeding number.

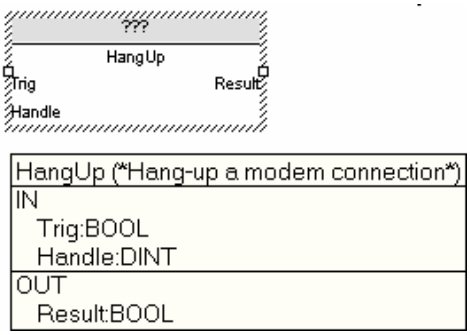
Phone: (STRING); One or more number, separated by ';'

Output parameters:

Result: (BOOL); True = success; False = some error, probably invalid handle, already dialing.

Remarks: There should be declared one instance of this FB for each modem driver (physical modem) used.

HangUp - Hang-up current connection



Input parameters:

Trig: (BOOL); '0' to '1' transition will start hang-up

Handle: (DINT); Protocol driver to hang-up. Result from one of the protocol driver configuration functions.

Output parameters:

Result: (BOOL); True = success; False = some error, probably invalid handle

Remarks: There should be declared one instance of this FB for each modem driver (physical modem) used.

GetCommState - Get communication status.

Called regular by STRATON application to get current status of the driver, and then determine which action should be taken. This function is primary intended for monitor data transmission on modem driver, but could be used by null modem drivers as well.

Input parameters:

Handle: (DINT); Modem driver to get communication state from. Result from one of the protocol driver configuration functions This is modem driver related, and different protocol handles at the modem driver will return the same result

Output parameters:

Result: (BOOL); True = success, return parameters are valid; False = some error, probably invalid handle.

State: (DINT); Bit 0..3

1. Idle	Driver is waiting for action.
2. Init	Initializing modem.
3. InitError	Not able to initialize modem (keep trying).
4. Dialing	Driver is dialing a number, and waiting for connect.
5. Con. Out	Outgoing call, Modem is successfully connected, driver is running.
6. Con. In	Incoming call, Modem is successfully connected, driver is running.
7. NoCon.	No connection. The dialed number was busy, did not answer etc. Waiting to redial.
8. Hang-up	Modem is hanging up the connection

Bit 4..6	Not used
Bit 7 Con. Err	Dial-up procedure failed. No connection was made to Main or any sub phone numbers. Dial suspended. The bit is reset when a hang-up and new dial-up is executed, or an incoming call is received.
Bit 8..31	Not used

Tlf Nr.: (DINT); Telephone number in the dial-up string, which are currently active.

HandShake: (DINT);

Bit 0	CTS	(in)	Clear To Send
Bit 1	DSR	(in)	Data Set Ready

Bit 2	RI	(in)	Ringing Indicator
Bit 3	DCD	(in)	Data Carrier Detect
Bit 4	DTR	(out)	Data Terminal Ready
Bit 5	RTS	(out)	Request To Send
Bit 6..31			Not used

RxByte: (DINT): Number of serial bytes received.

TxByte: (DINT): Number of serial bytes transmitted.

RxFrame: (DINT): Number of protocol frames successfully received. Used to detect communication is running. In modem mode the counter is reset to zero when line is hang-up.

TxFrame: (DINT): Number of protocol frames successfully transmitted. Used to detect communication is running. In modem mode the counter is reset to zero when line is hang-up.

How to use the modem dial functions may be better understood if you get one of our available STRATON examples. Check your RTU32 CD or our homepage for useful examples.

5.8 Data Logging

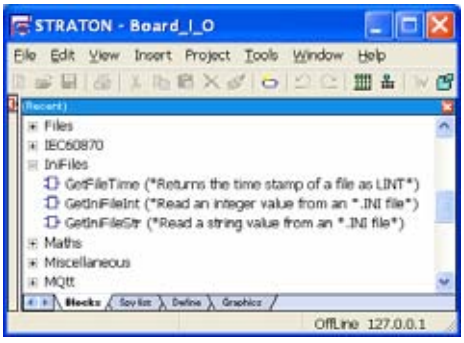
STRATON provides functions for logging data into a file in the RTU32. In general 2 functions are available:

1. Logfile CSV – simple function for logging data into af CSV-file.
2. Read/write functions – to use for configure almost any extra log function into a file.

See STRATON help for details.

5.9 Reading Text Files from STRATON

To provide possibility to read variables and string from an external file in the RTU32, a few simple read file function is supported. The functions are found in the IniFiles folder.



You have to place the text file in the RTU32 file system manually – use FTP to do this. MS Internet explorer can be used to explorer the RTU32 files. Use FTP instead of HTTP in the address field, like: ftp://192.168.100.243/. An empty folder will come up – right click in it and login using your username and password (same as used for entering the web pages).

The file structure must be as the SNMP.txt and RTU32.ini. The structure is simple and looks like this:

[ABC]

def=123

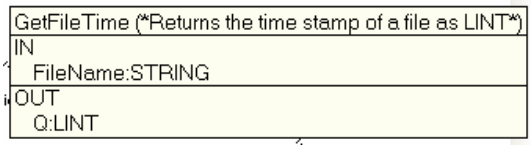
ghi=456

ABC is called the section and def the item. NOTE: all is case sensitive.

The functions available are;

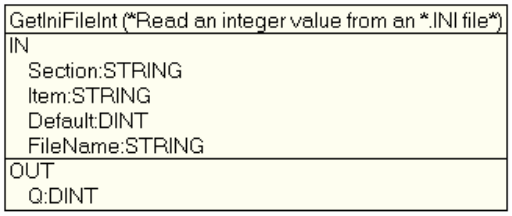
- GetFileTime

  With this function you can check the time of the file so you only read the variables in case of changes.



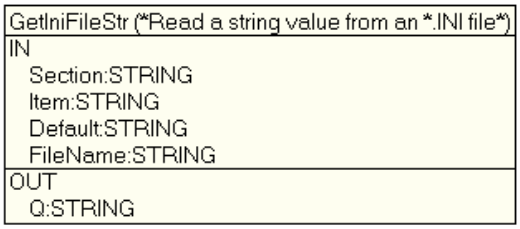
- GetIniFileInt

  Read an Integer value from a section and item.



- GetIniFileInt

  Read a String variable from a section and item.



The idea with these functions is that you can place all your variables outside STRATON application program. You can e.g. use same application program for several sites and only have the differences in the file.

You can edit the file from the web editor with your browser, use the RTU32TOOL editor or directly send a new file to the RTU32 with FTP.

More help is found in STRATON by pressing F1.

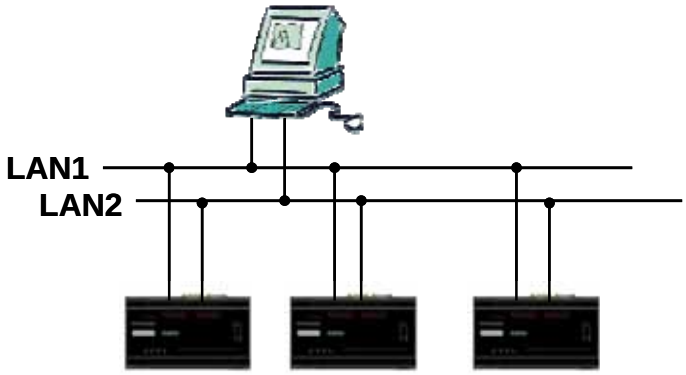
5.10 Event Binding Protocol / Redundancy Dual Binding Communication

General

The STRATON Binding protocol is a proprietary protocol for fast time stamped event based data transfer directly from one STRATON runtime to another. The binding protocol is communicating directly from one RTU32 to another RTU32 in a LAN/WAN network. It can run disregard less of any other application. In a network with several RTU32s you can with a Global Binding Editor bind any global declared variables together.

The same can be used between a STRATON PC runtime and an RTU32.

Dual binding is used in redundancy network connection using 2 separate network segments. Both the unit publishing variables (producer) and the unit subscribing variables (consumer) has to be equipped with 2 Ethernet network interfaces. Each network connection must be setup on the units as 2 separate network segments using different TCP/IP subnet masks. The compiler also takes care of handling the sorting out of timing issues of any changes in values. This means that you on the subscriber side always will get the correct value. The first received event will be reported at the subscriber side and the same event with the same time stamp received secondly will be ignored.



Network settings in RTU32 (dual binding)

If you want to run dual binding using the RTU32 you must configure both Ethernet interfaces. No additional settings are necessary as the networks as seen as direct accessible

parallel networks. Routing to the 2 networks as done as default. If you ping an address on any of the 2 selected network segments, the ping request will find its way to the correct segment automatically.

Procedure for setting up binding/dual binding in STRATON

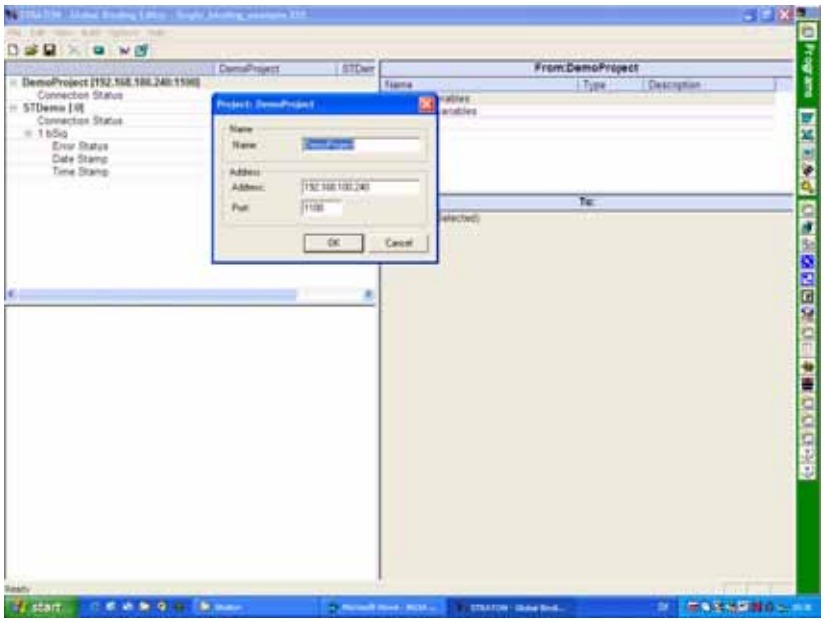
You need to program all the RTU32 (and PC with STRATON runtime) before starting to bind (link) together the I/Os. I/Os and variables to bind must be declared as Global variables.

After making the program you must start the Global Binding Editor (find it under Start/Programs/STRATON/.

The dual binding option is configured in the Global Binding Editor. See Global Binding Editor Help for general details of using the Global Binding Editor.

To add a redundancy communication network channel in the editor, you must enter 2 IP addresses of each variable you publish and subscribe.

Below figure shows the configuration window which will be added with additional configuration fields for an IP and port address.



If you define 2 IP addresses (IP1/IP2), the compiler will automatically setup to use the two communication channels for parallel event reporting of events.

5.11 Optional SNMP Extension Agent Driver

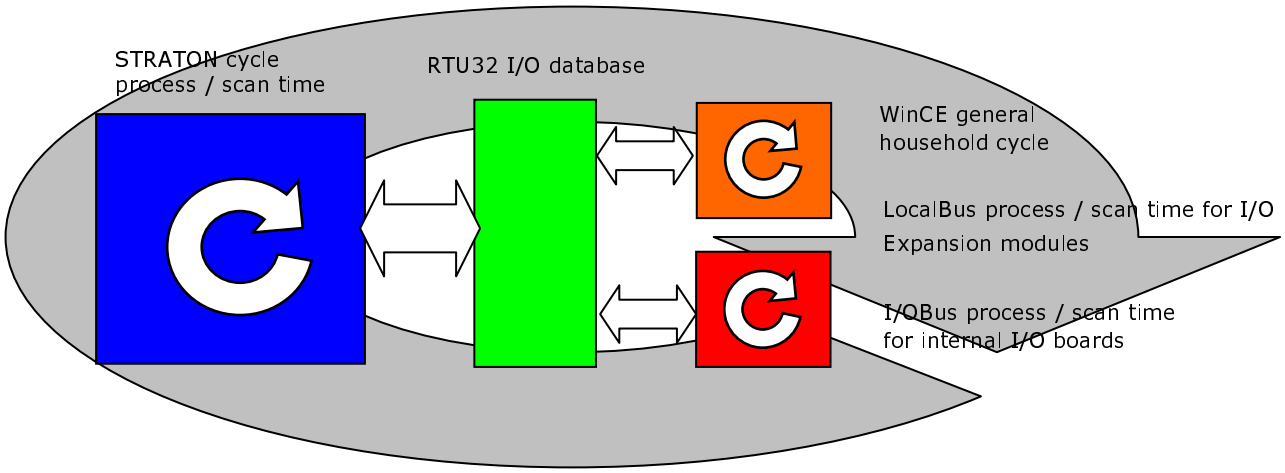
The RTU32 can be delivered with support for SNMP. Agent Driver used for Network Alarm and management applications. The SNMP Agent Driver provide SNMP protocol functions for Trap Alarming, Get/GetNext and Set Command. All features are fully configurable in the STRATON application program. See the SNMP driver manual for details.

6. RTU32 cycle / scan time

6.1 General

The RTU32 does in basic these include 4 processes which is relevant when determine timing performance for a RTU32;

- STRATON EN/IEC61131 application program cycle/scantime
- LocalBus cycle for expansion I/O modules
- Internal I/OBus cycle/scantime
- WinCE general household cycle



Each cycle are running fully independent (asynchronous) from each other and must be considered without internally synchronisation. The cycles does however have influence on each other as they all do load the same CPU and they are also assigned different priority in order to obtain the best possible overall performance. In general are processes for STRATON application program and I/O handling running in high priority threads in WinCE.

The LocalBus driver is internally communicating with a sub-processor, which is managing the low level LocalBus driver. It is running as fast as possible and the scan time is directly dependant on the number of connected LocalBus I/O Expansion modules. The LocalBus are reading/writing I/Os and errors on the I/O modules. The LocalBus itself is a simple but robust shift register bus.

Internal I/OBus is managing the scan for the built-in I/Os in the RTU32. It is providing fast I/O updates because it uses an internal parallel I/O bus communication.

The STRATON program cycle is executing the actual application program developed in the STRATON WorkBench. It is also running in a high priority tread. The STRATON developing environment provides several functions to control and optimise the application program processes. Refer to STRATON Training course for details.

6.2 RTU32 Scan cycle mechanisms

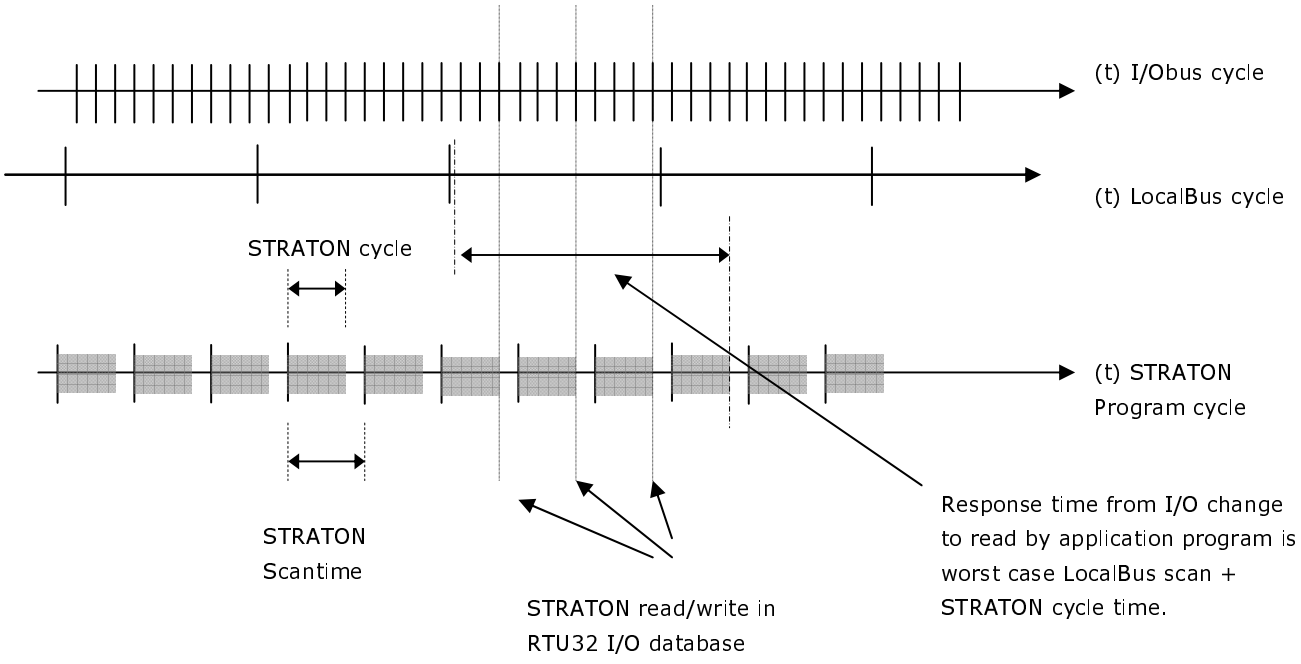
The 3 first tasks described above do run independent of each other.

The 2 I/O cycles are running as fast as possible to ensure that the basic RTU32 I/O database at any time reflect the physical I/Os – both internal I/O and I/O on the expansion modules connected.

The STRATON total program cycle time is depending of the size of the application program.

At the end of every STRATON cycle the I/Os defined in the application program is read/written to the basic RTU32 database. The STRATON scan time (time between each cycle) can be adjusted – see section STRATON scan time section for details.

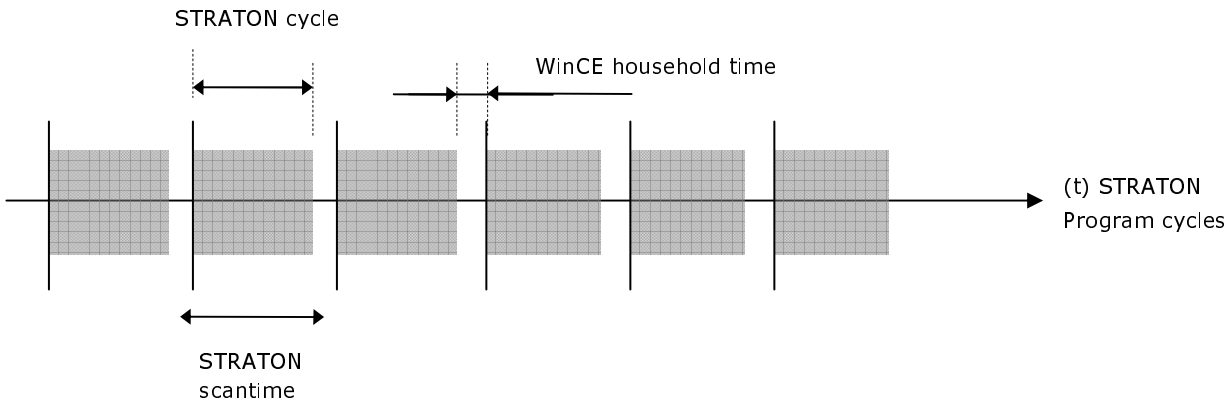
The link timing between these processes can best be explained in a graphical way.



Note: The timing in the figure is shown to explain the link between the cycles. The timing of the cycles in a real application will be according to the actual STRATON application program size and number of connected I/O expansion modules.

6.3 STRATON scan time

The STRATON scan time can be set to 0 (zero) which automatically assign some time for operating system household between each STRATON program cycle. In this case the program runs as "fast" as possible. This is however not all correct. The automatic assignment of household execution time leaves relative much time for household to ensure all necessary operating system processes can be maintained with a good margin.



If you set the STRATON scan time to a fixed value, you can in some cases obtain a better general scan time for your application program. But it is IMPORTANT to realise that you are then responsible to leave time for household operations.

IMPORTANT NOTICE: If you leave to short time for household operations, you will have overrun in your STRATON program which gives fatal errors. And it require verification test to ensure that a fixed scan time under all conditions do not cause overrun!

To give a picture of the general STRATON application scan time performance, please see appendix 1 with some example benchmark measurements.

LocalBus scan time

The scan time of the local bus to Expansion I/O are running as fast as possible and is directly depending of the number of connection I/O Expansion modules. Data communication is a simple and reliable shift register bus.

The procedure for the communication is that the I/O modules are initialised at boot up. After that the internal I/O database in configured and the I/Os are started scanned. Cyclically the I/O configuration is checked for errors.

As a rule of thumb the following calculations and scan time examples can be used;

The maximum number of either input, or output connected determinate the scan time.

I.e. 16 x UCL-16DI configuration will have the same scan time as 16 x UCL-16DI, 16 x UCL-16DO.

As a rule of thumb the scan time could be calculated as 6 ms + number of modules (16 DI, DO) * 0,25ms

Configuration	**Scan time *)**	**Calculation**
1 x UCL-16DI	5..7 ms	(6 + 1 * 0.25)
16 x UCL-16DI	10..12 ms	(6 + 16 * 0.25)
32 x UCL-16DI	14..17 ms	(6 + 32 * 0.25)
32 x UCL-32DI	20..24 ms	(6 + 32 * 0.50)
32 x UCL-16DO	14..17 ms	(6 + 32 * 0.25)
32 x UCL-08AI	14..17 ms	(6 + 32 * 0.25)
16 x UCL-16DI, 16 x UCL-16DO	**)10..12 ms	(6 + 32 * 0.25)

*) As it is indicated the scan time varies during scan. That is caused by the cyclic configuration control of the I/O modules which is done occasionally.

**) The scan time reflect scanning of one analogue input or output on an analogue expansion module. As the analogue input expansion modules do include several inputs or outputs which are multiplexed, you must multiply the scan time with numbers of input/output to get the real scan time.

Example: If you use one UCL-08AI.D1 Expansion module with 8 AI on a RTU32, you must expect the scan time for each of the inputs to be 8 x (6 + 1 x 0,25) = 50ms. In other words it needs 8 total scans before all 8 AI on the modules has been read.

Internal I/OBus scan time

The scan time of the RTU32 internal I/O are running as fast as possible. The communication is handled via mapped I/O in a sub CPU. The scan time is fixed and independent of any other tasks running in the RTU32, i.e. it has no influence how many I/O expansion modules that are connected to the RTU32.

The procedure for the communication is that the I/O board is initialised at boot up. After that the internal I/O database in configured and the I/Os are started scanned.

As a rule of thumb the following calculations and scan time can be used;

Configuration

I/O board configuration 26IO with 16 digital inputs (also working as counter inputs), 4 relay outputs, 4 configurable analogue inputs and 2 configurable outputs.

I/O type	**Scan time**
Digital inputs/outputs	5 ms
Analogue inputs	150 ms
Analogue outputs	50 ms

The differences between digital and analogue update time is caused by sampling time in analogue converters and the multiplexer. Also signal stabilizing in the electronic analogue circuits are contributing time to the scan time.

7. STRATON HMI - Getting started

Before you start

The HMI function in RTU32 and STRATON WorkBench is based on ActiveX. It means that you will require an ActiveX components running on your PC. It is automatically downloaded when access the RTU32 HMI webpage. The ActiveX also require an application file. This is important that you install these STRATON OCX files before you start. You can install them from;

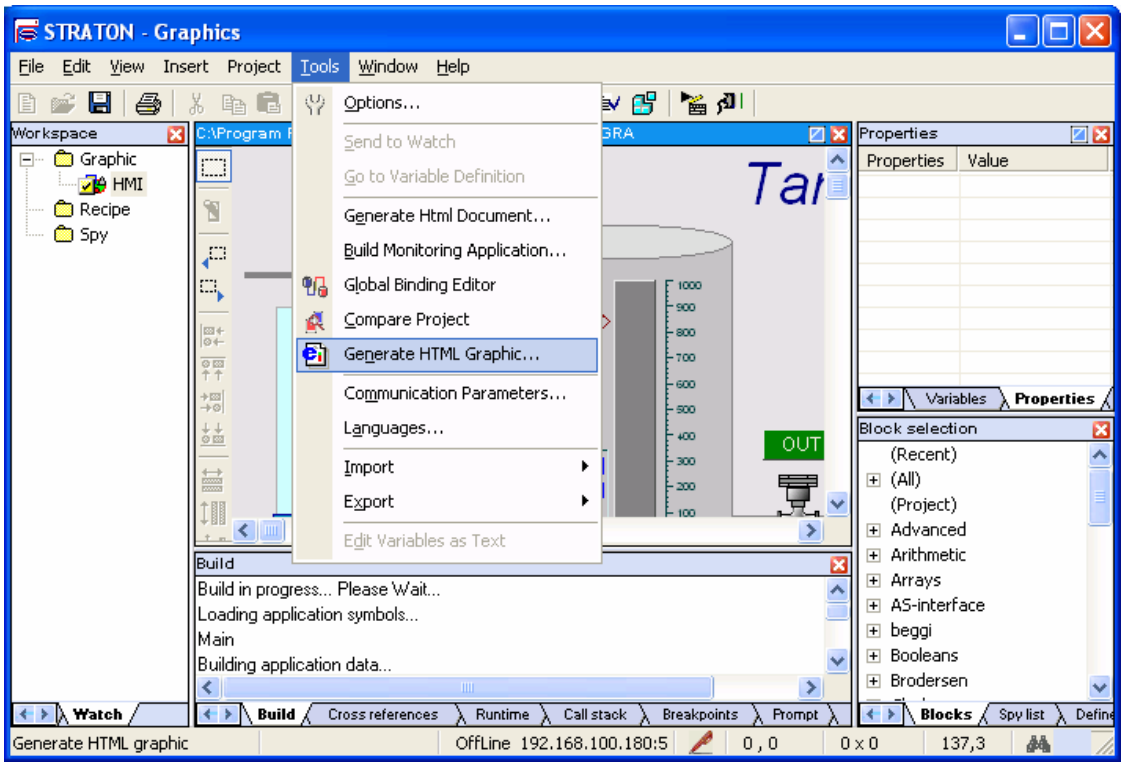
http://www.copalp.com/download/instal/sr6/SetupOCX.exe

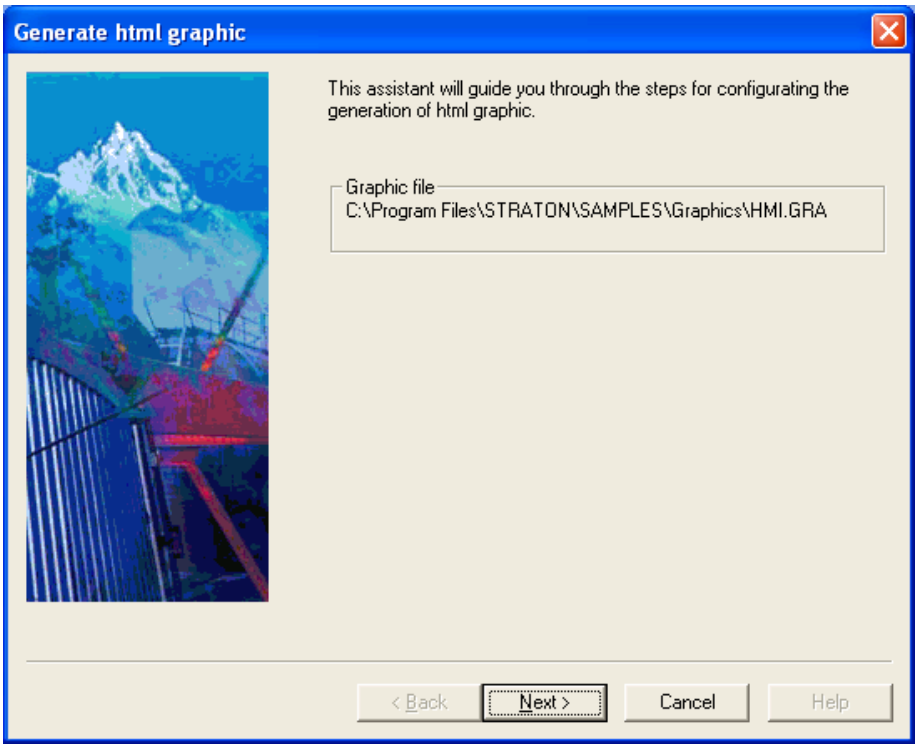
or from the RTU32 CD

The HMI function is supported from STRATON WorkBench version SR6 or newer and in the RTU32 general software version BC1.00 / CE1.00.17.

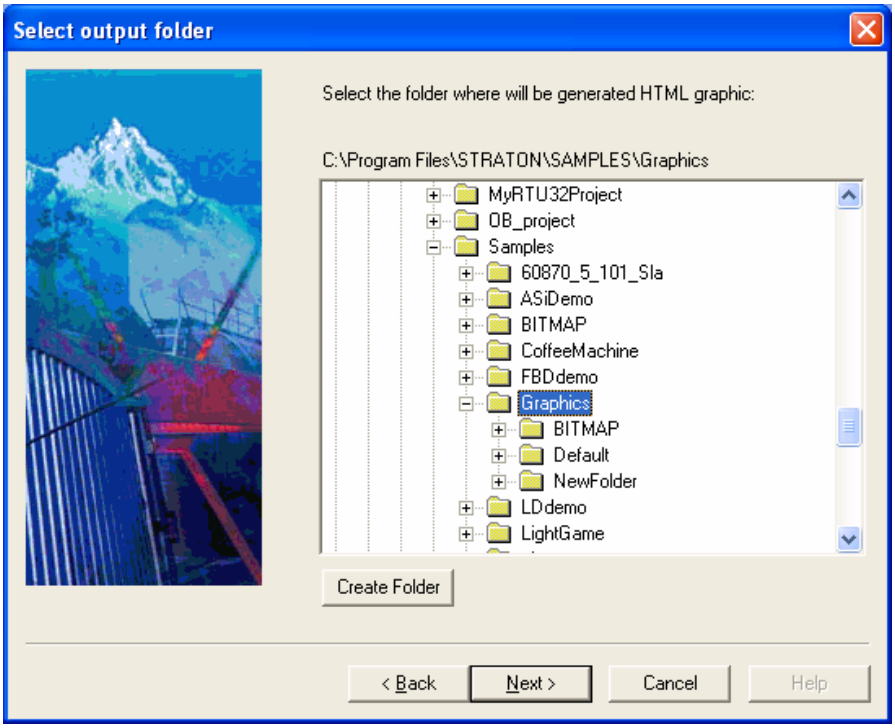
HMI Procedure

Create a simple graphic project – Download and run it on RTU32. Go offline and then select "Generate HTML…" as shown below.

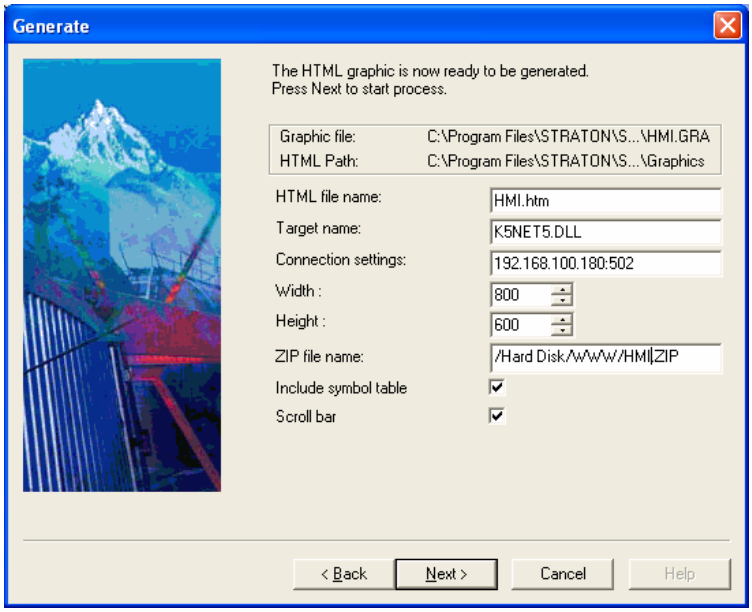




Then this window pop up - select “Next”.

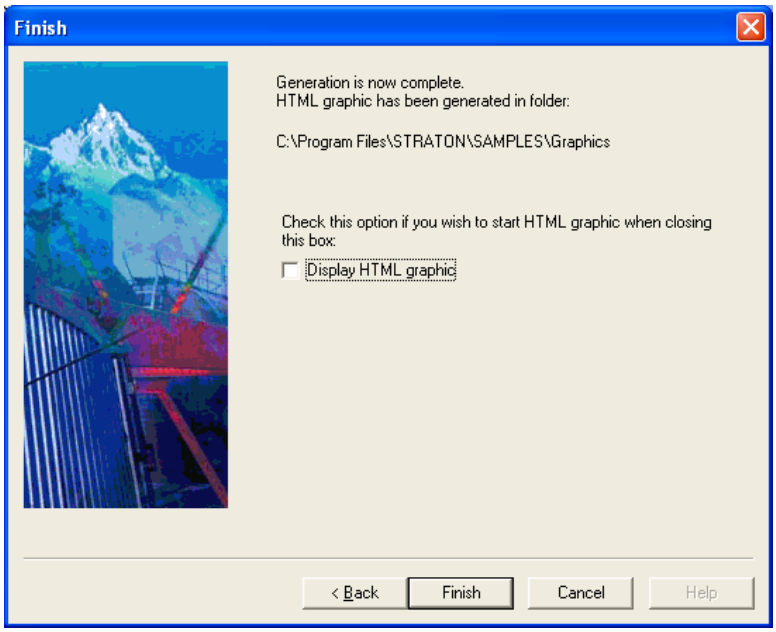


Here you MUST select an empty sub-folder. Create e.g. a new one called HMI and the select "Next"



Enter the parameters as shown. The HMI.htm is always the name of the web page on the RTU32. The Connection settings should be the same address as you use for the general STRATON Project. And the ZIP file name must include the path and always have the name HMI.ZIP.

When entered then select "Next" to continue.



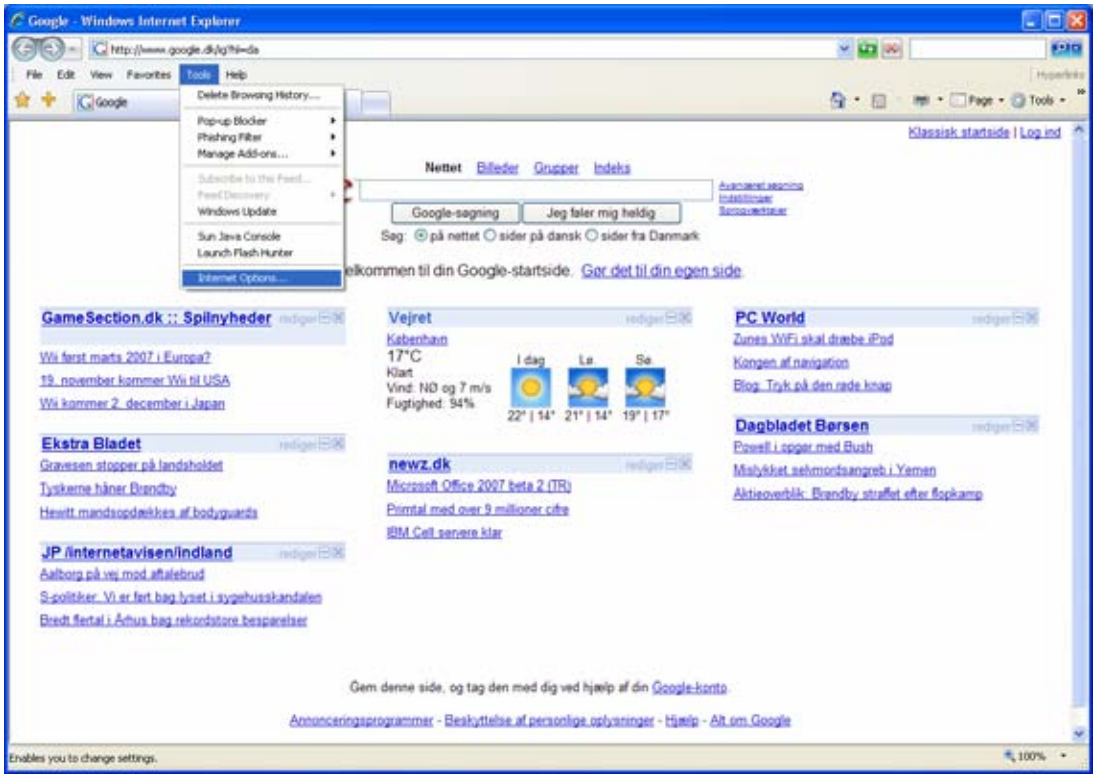
Uncheck "Display HTML graphics" and select "Finish" to finalise.

Then 2 pop-up boxes are shown very fast – it indicates that the HTML page is downloaded to the RTU32.

Note: If you check the RTU32 file system with FTP, you will see 2 new files; hmi.zip and hmi.htm in the ./Hard Disk/WWW/ folder.

The HMI part is now ready for monitoring the RTU32 HMI page with a Browser supporting ActiveX. We recommend using MS Internet Explorer.

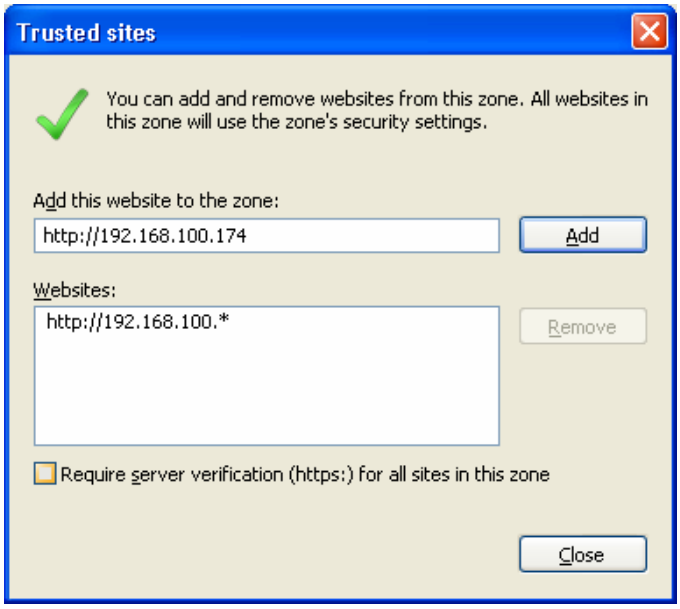
Note: *As your Internet Explorer have several built-in security functions, you should add the whole local network IP segment or only the RTU32 IP address as trusted.*



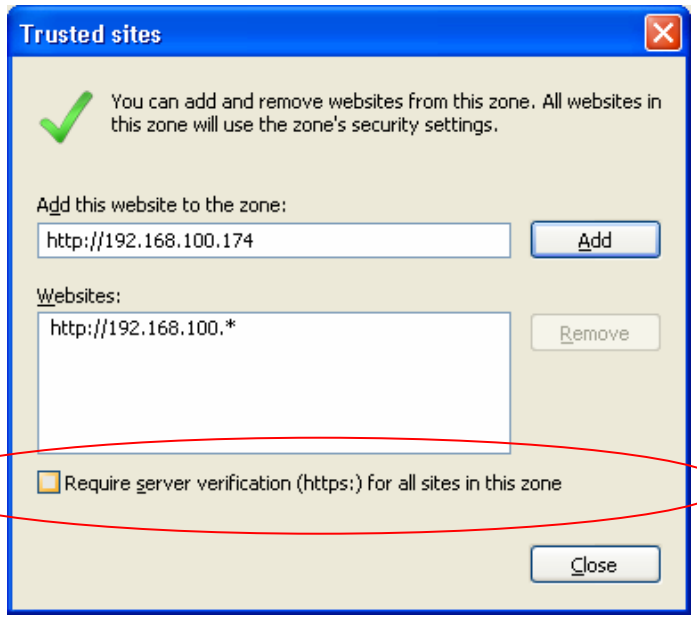
In the internet options select the "Security" bar. Now select "Trusted Sites" and click on the "Sites" button.

Enter the IP address of the RTU or the IP address of the local network ending with "*" to add the whole local network as trusted source.

By adding the whole local network as trusted source/site, you will avoid repeating this process for other RTU's on the same IP segment.



Note: the "Require server verification (https:) for all sites in this zone" field should not be selected.



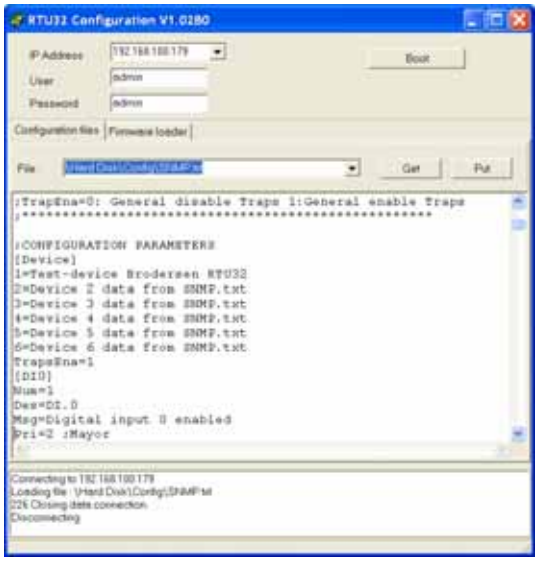
8. RTU32 Utilities and System Event Log

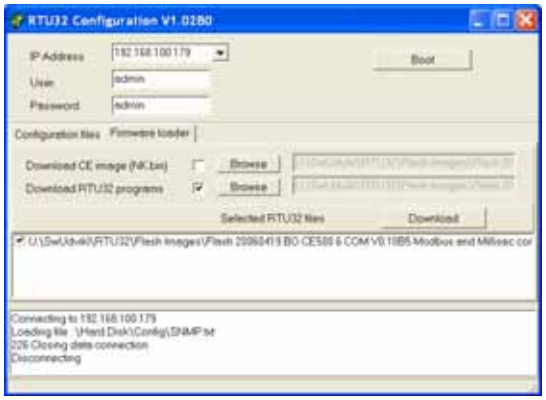
8.1 General

To help you Brodersen provide some utility programs for RTU32. RTU32TOOL for RTU32 maintenance and configuration and EventViewer for RTU32 system log monitoring.

8.2 RTU32TOOL

The RTU32TOOL is the general RTU32 software tool developed for remote maintenance and configuration.





In the first version are functions for remote update of RTU32 software/firmware and an editor for the RTU32 configuration files. The remote update function are only to be used over LAN networks and provide the function for a totally update of the software in RTU32. The update includes WinCE operating system, Brodersen specific RTU32 drivers and the STRATON runtime part (VM). The update can be done in total or partly – you select the files to be updated.

The update function uses FTP for file transfer to the RTU32 and includes an intelligent function for saving the basic network setting during update. With this function the network connection settings are kept – and you are able to obtain the connection to the RTU32.

The file editor is used for editing RTU32 configuration files. All configuration files are placed in the /Hard Disk/Config/ folder on the RTU32. At the editorial time only a configuration files are default in the RTU32. It his however you option to add your own txt files into this folder and use your STRATON application to read all predefined variables from this file. It means that you can edit the variable stored in the file without using STRATON WorkBench.

In the STRATON WorkBench you will find the necessary function blocks for reading file data.

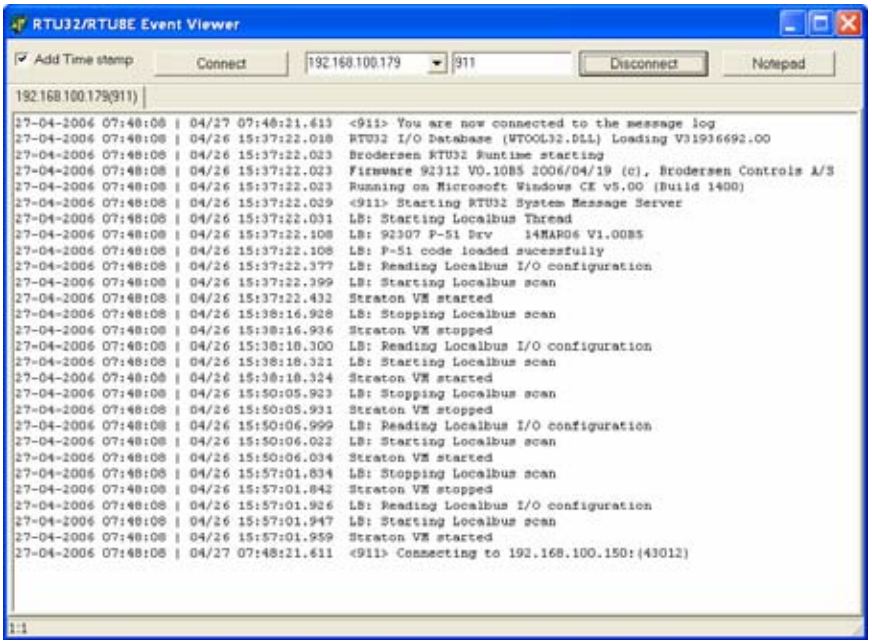
See Reading text-files section for details.

8.3 System Log/Event Viewer

The system log provides you with information about started processes in the RTU32, errors and status information's. You can furthermore add debug or information texts in your STRATON application program which will be printed in the system log. COM port communication can be logged if you add the right parameters to the COM port setup – see COM port settings for details.

The System Log can be viewed by using the Brodersen EventViewer utility or a standard Telnet window. The EventViewer is a simple Telnet utility.

The System Log is available on port 911 and look like this in the EventViewer;



The RTU32 allows max 20 continuous connections and the log FIFO include max 100 messages.

9. RTU32 Technical Description

9.1 General

The RTU32 Outstation and Industrial Controller are based on an Industrial PC platform running WinCE operating system with all the well known embedded Microsoft environment facilities. An industrial power supply is integrated to support industrial supply levels. The basic IPC includes a range of communication and other interfaces. The RTU32 provides additional interfaces like LocalBus for expansion I/O and integrated I/O.

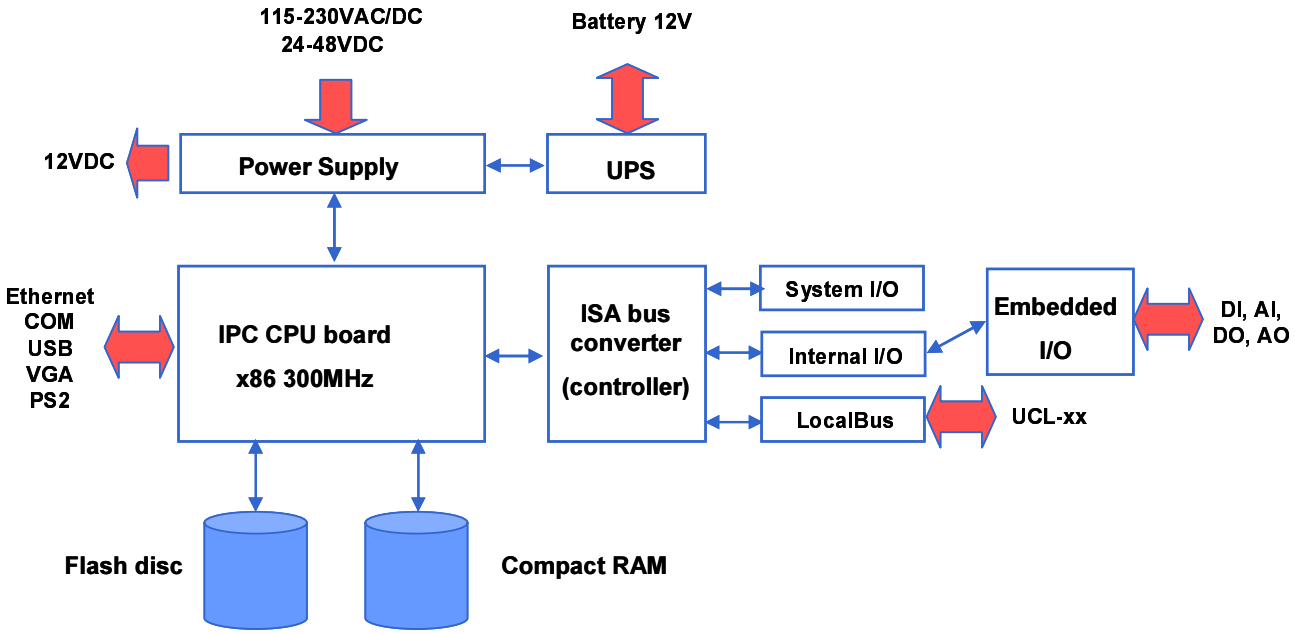
The RTU32 software is stored on a removable Flash. During start-up, the operating system and applications are moved to RAM where it is executed. Retained variables are stored executing a special command in the application program.

WinCE is the RTU32 real time operating system. Integrated I/O and LocalBus for external I/O connectivity are controlled in an implemented I/O database. A STRATON SoftPLC VM (Virtual Machine) is ported to the WinCE real-time task. This enables the STRATON SoftPLC runtime application program to be executed in the RTU32.

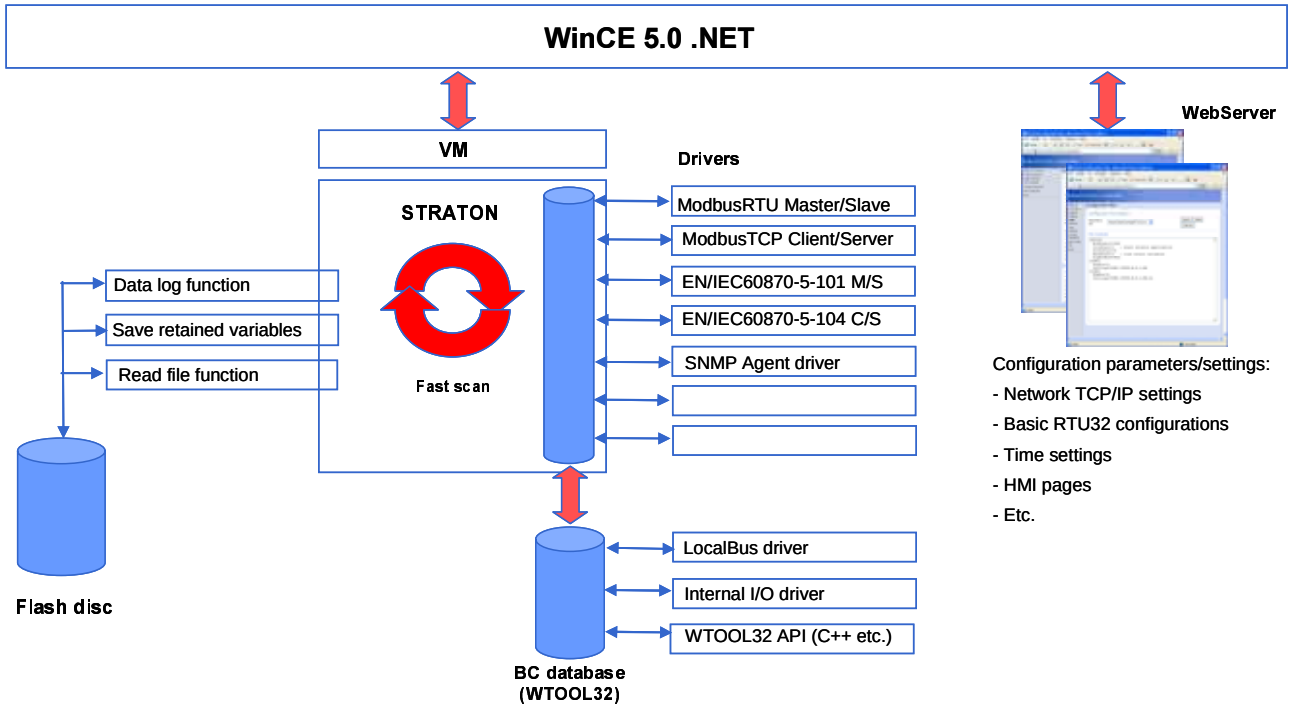
Using the Ethernet network for primary communication provides all the advantages of existing TCP/IP networking communication facilities. Fast, reliable and secure communication is the main features and all networking components (software, routers, switches, etc.) are available. In addition, serial ports for interfacing to application specific protocols (e.g. Modbus, Fieldbus, utility and traffic proprietary protocols etc.) are available.

Detailed technical information about the RTU32 hardware is provided in the RTU32 Data Sheet and in the RTU32 Wiring and Mounting Instruction manual.

9.2 Hardware block diagrams



9.3 Software block diagrams



9.4 STRATON SoftPLC

The STRATON programming tool fully supports EN/IEC61131 and is used for making SoftPLC programs in the RTU32. The application program kernel is implemented and runs in WinCE real-time task. STRATON offers complete SoftPLC functionality and supports all features needed in today's industrial environment. STRATON supports programming languages such as Structured Text, Function Block, Ladder, Instruction List and Sequential Function Chart. The STRATON SoftPLC supports cold restart, hot restart and on-line changes.

The STRATON Workbench is used for configuration protocols, programming and debugging. It supports several tools for multi-program handling and documentation. It is also a powerful tool for complete system design and programming, providing unique functions for event based communication. The Global Binding Editor makes it possible to publish and subscribe variables in a large network with minimum communication load. The events are time stamped and can also be used directly in ZenOn SCADA HMI applications.

Programming, debugging and upload and download of application programs can be done remotely via Ethernet or RS232.

9.5 RTU32 LocalBus for I/O Expansion

General

The LocalBus is physical a simple string of shift register, where data is clocked serial, and latched parallel to I/O. The raw data is shifted to / from two data buffers, and processed by type ID dependent functions into data type sorted IO-data structure. The IO-data structure operate with the data types as defined in the STRATON I/O Driver.

LB driver

The LocalBus driver is running cyclic in a high priority thread, and always started at RTU32 boot time. However start and stop of LB scan is controlled by STRATON application.

When started, LB configuration is read, and database is build accordingly. The driver has knowledge of every module type layout, and builds an IO-data structure which matches the physical connected modules. If a module of unknown type is connected, the IO-data structure could not be build, LB scan is not started, and an error is reported in the system event log.

After reading the LocalBus configuration, a copy of the IO data structure is also saved on the flash. The reason for that is to support a lock function which freezes the LocalBus configuration. If the lock is enabled the configuration read at STRATON application start are compared to the saved configuration on the Flash. In case the configurations do not match, the local bus scan is not started and an error is reported. The lock is implemented as a configuration option in the RTU32.ini file; LockIOconfig=0. In case the lock function is disabled (default), the configuration is just copied to the Flash.

The LocalBus driver has the limitation of max 32 external I/O Expansion modules.

LB driver event information

Information and status of the I/O drivers are reported in the system event log read at port 911. The information about the LocalBus driver is identified by "LB:" and include information and status like;

```
Starting LocalBus Thread
Firmware 92307 V1.01B3   2006/06/08
Code loaded successfully
Reading LocalBus I/O configuration
I/O board read: 26IOA
I/O board read: 16DIO
I/O board read: 8AI.Dx
Starting Localbus scan
```

If the I/O configuration changed while the RTU32 is running or changed whiled locked, it will be reported in the log like;

```
Localbus error, I/O configuration mismatch
```

If you have locked the I/O configuration, and the physical I/O configuration is not correct you will get a report in the Event log. It will tell you what configuration was expected and what is actual read.

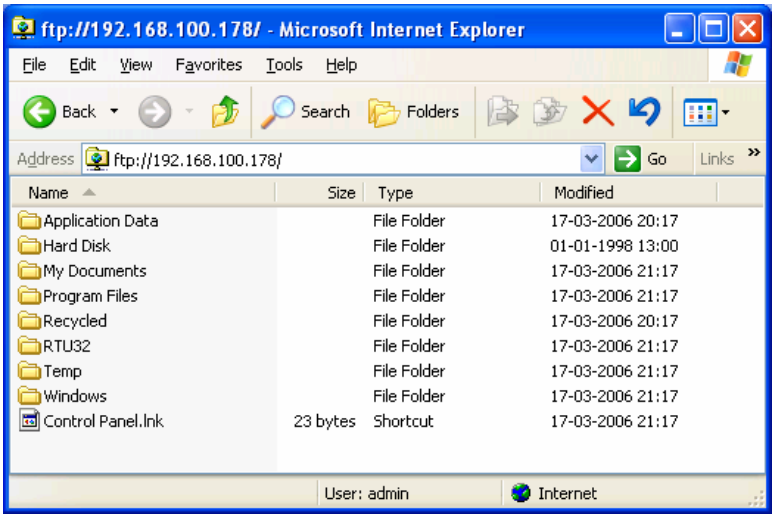
9.6 RTU32 File System

General

When you apply power to the RTU32, it will boot from the flash disk. All necessary files for OS, STRATON kernel, drivers, application program etc. are located on the flash disc.

During boot the OS is moved to run on the internal RAM. The files located on the flash disc will be moved to the /hard disk/folder.

If you access the running RTU32 with FTP you will see the root directory like:

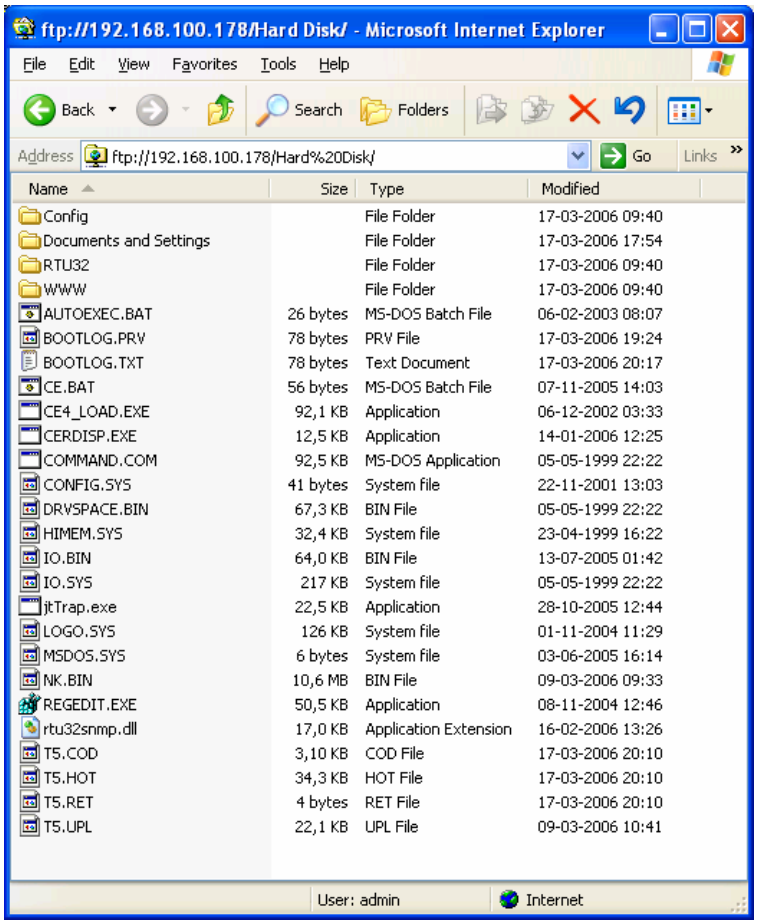


The root directory contains a number of subfolder and system files.

What does the folders contain?

../Hard disk/ folder

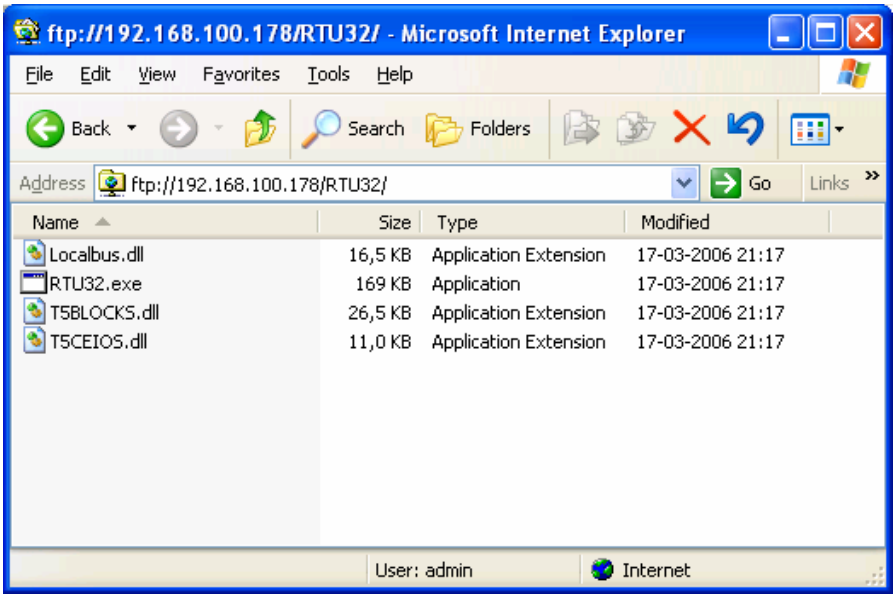
Contain all the files which are actually on the flash disk.



If you want to save files to the Flash in the RTU32, they should always be saved in the ../Hard Disk/ folder.

./RTU32/ folder

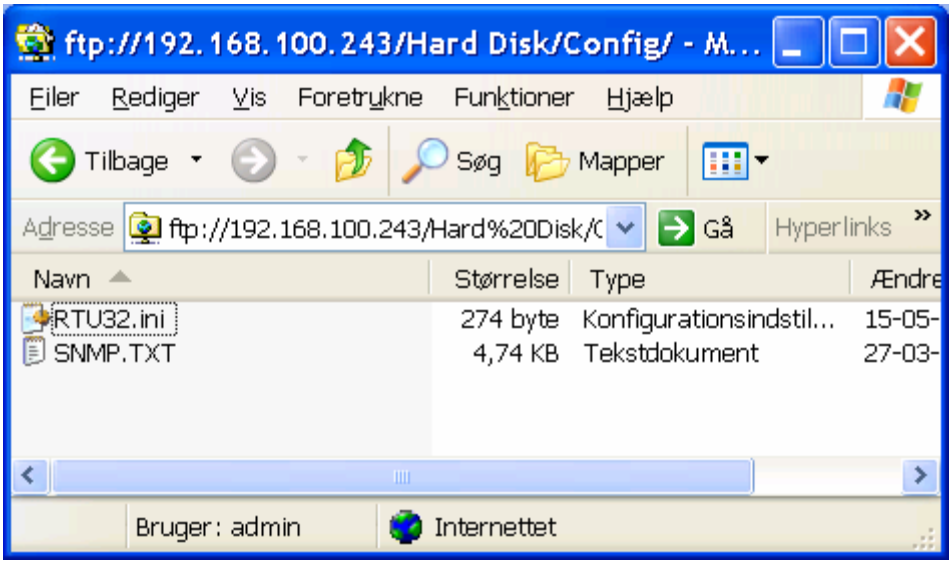
Contain the RTU32 specific DLL driver files.



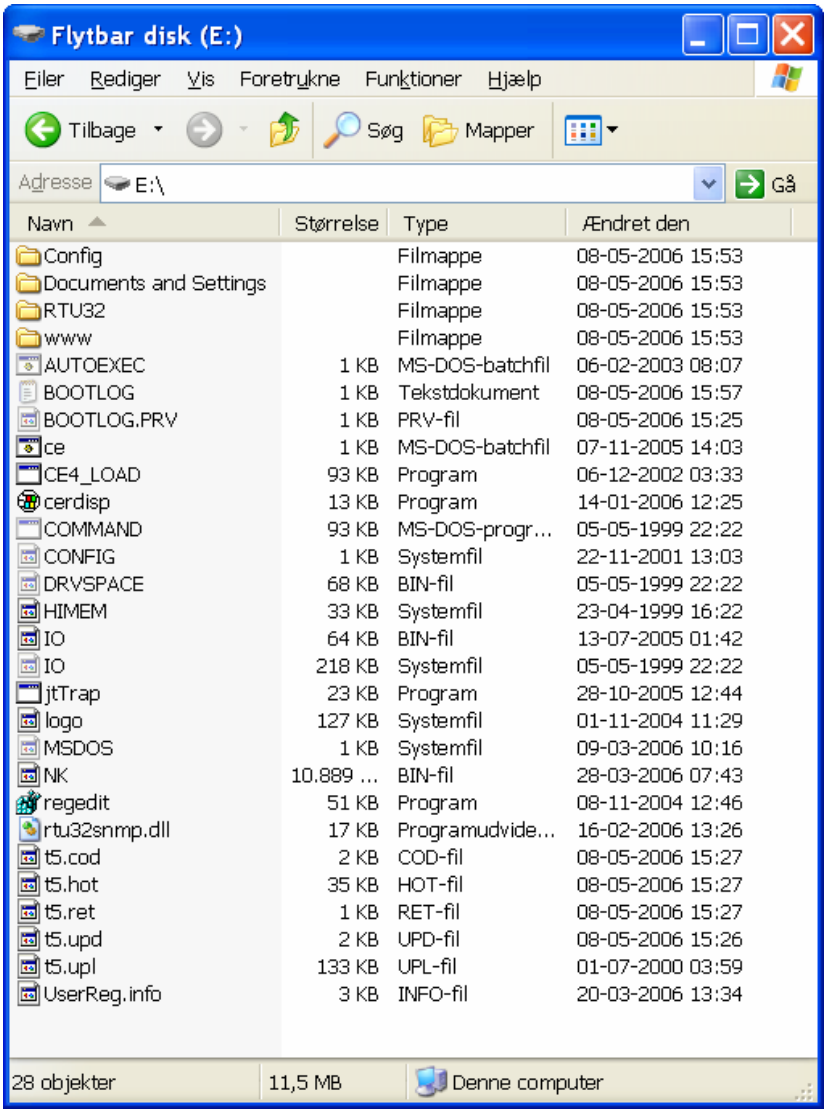
The RTU32.exe file is the actual virtual machine file for executing the STRATON application program.

../Config/ folder

The folder contains the configuration files. The RTU32.ini file defined the basic RTU32 configurations. The SNMP.txt file is used by the optional SNMP Agent driver. If you are going to use the read text file function in STRATON, the file should be placed in this folder



Flash Disc Files



The files on the Flash disc contain all the files necessary to run the RTU32. As mentioned before the files equal the ../Hard Disk/ folder you will find on a running RTU32 using FTP.

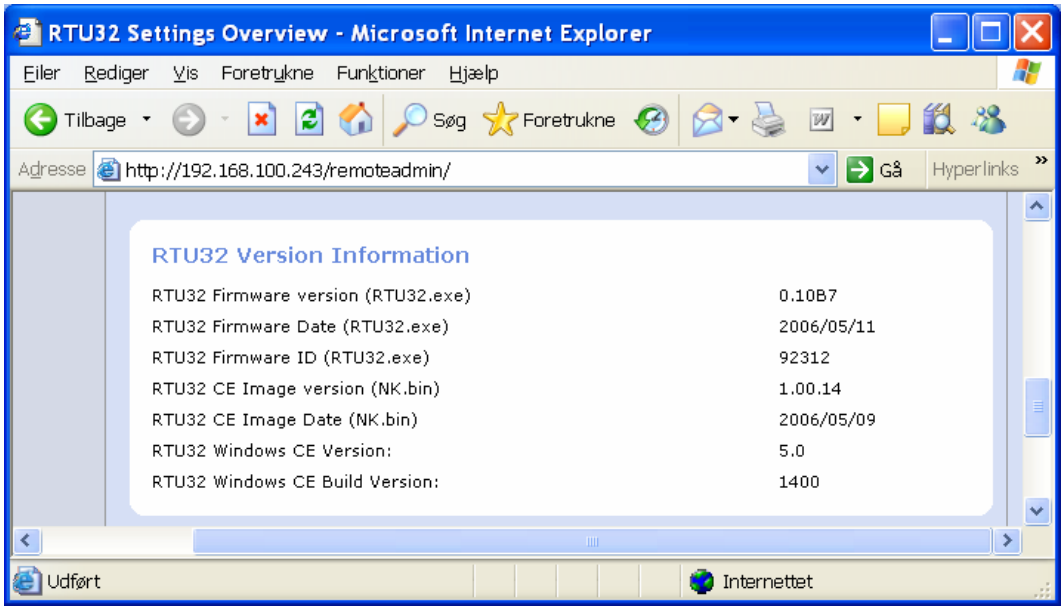
The NK.BIN is the Windows CE image file.

The t5.* is the STRATON runtime files. If you by any reason have a problem with a application program freezing the RTU32, you can delete these files and the runtime application deleted. You can now download a new application from the WorkBench.

The /Documents and Settings/ folder contain all you special settings for Windows CE – including network setup etc. If you delete this folder and all its content, the RTU32 will start-up as a virgin RTU32 (as delivered from the factory)

9.7 Software Versions

The Settings overview page in the RTU32 includes the software version for the actual build.



The software in the RTU32 includes 3 main software parts;

- RTU32 Firmware
- Windows CE image
- Windows CE platform builder

RTU32 Firmware include the specific RTU32 functions and do include all files in the ../RTU32/ folder on the Flash. The RTU32 firmware is assigned an id (e.g. 92312), a date and a version

RTU32 CE Image is the single NK.BIN file is assigned a date and version.

Windows CE version is the MS Platform builder and is defined by a general version and a build version.

If you contact Brodersen or your local distributor / system integrator for support, it is important that you have these software information's ready together with the serial number of your device(s).

Appendix 1

RTU32 STRATON applications measurement / benchmark

The RTU32 STRATON runtime cycle time is measured on a number of different applications. Small and larger applications, with and without serial communication. The RTU32 scan time is in STRATON set to 0, which means that STRATON automatically assign time for household. This setting is recommended as it ensures that your application will avoid any overrun situations.

Example 1, STRATON application program without any additional serial communication

Small test configuration:

1 x RTU32 without internal I/O

1 x UCL-16DIO Expansion module

1 x UCL-08AI Expansion module

2 x UCL-04AO Expansion modules

All 8 analogue inputs are scaled, and some simple math executed, before stored in an internal register, and copied to an analogue output. (Approx. 15 STRATON instructions pr channel).

10 digital inputs are controlling one timer each, which output is connected to a digital output.

6 digital inputs are controlling 6 counters, which output is connected to a digital output.

Result:

The typical STRATON scan time is 2ms.

Large test configuration:

1 x RTU32 without internal I/O

6 x UCL-16DIO Expansion modules

6 x UCL-08AI Expansion modules

12 x UCL-04AO Expansion modules

All 48 analogue inputs are scaled, and some simple math executed, before stored in an internal register, and copied to an analogue output (approx. 15 STRATON instructions pr channel).

40 digital inputs are controlling one timer each, which output is connected to a digital output.

30 digital inputs are controlling 30 counters, which output is connected to a digital output.

Result:

The typical STRATON scan time 4ms.

Example 2; STRATON application program with Modbus communication

A number of tests are executed on the STRATON build in Modbus slave / master driver, using the Fieldbus configurator.

Small test configuration:

1 x RTU32 without internal I/O

1 x UCL-16DIO Expansion module

1 x UCL-08AI Expansion module

2 x UCL-04AO Expansion modules

All 8 analogue inputs are scaled, and some simple math executed, before stored in an internal register, and copied to an analogue output. (Approx. 15 STRATON instructions pr channel).

10 digital inputs are controlling one timer each, which output is connected to a digital output.

6 digital inputs are controlling 6 counters, which output is connected to a digital output.

ModbusRTU Slave

Modbus RTU slave on COM1 9600 baud. 50 words are read/write in each frame.

Modbus TCP slave. 50 words are read/write in each frame.

Result:

There is no noticeable change in STRATON scan time, whenever communication was used or not. STRATON scan time is approx. 2ms.

ModbusTCP Client

Modbus TCP master. 50 words are read/write in each frame.

Result:

There is no noticeable change in STRATON scantime whenever communication was used or not. STRATON scan time is approx. 2ms.

Example 3; STRATON application program with additional STRATON binding protocol

RTU32 is running the standard application, and using binding protocol to transfer events to another RTU32.

STRATON scan time is measured on the following applications.

Small test configuration:

1 x RTU32 without internal I/O

1 x UCL-16DIO Expansion module

1 x UCL-08AI Expansion module

2 x UCL-04AO Expansion modules

All 8 analogue inputs are scaled, and some simple math executed, before stored in an internal register, and copied to an analogue output (approx. 15 STRATON instructions pr channel).

10 digital inputs are controlling one timer each, which output is connected to a digital output.

6 digital inputs are controlling 6 counters, which output is connected to a digital output.

Binding setup

16 digital points (BOOL) and 8 analogue points (DINT) are transferred to another RTU32 every 1sec. The points are internal memory points, and are all changed in the same STRATON cycle.

Result:

The typical STRATON scan time is 3ms.

If data transfer are stopped (change of binding variables are stopped), STRATON scan time is still typical 3ms. I.e. there is no noticeable (less than 1ms.) change in performance if events are generated or not.

Large test configuration:

1 x RTU32 without internal I/O

6 x UCL-16DIO Expansion modules

6 x UCL-08AI Expansion modules

12 x UCL-04AO Expansion modules

All 48 analogue inputs are scaled, and some simple math executed, before stored in an internal register, and copied to an analogue output (approx. 15 STRATON instructions pr channel).

40 digital inputs are controlling one timer each, which output is connected to a digital output.

30 digital inputs are controlling 30 counters, which output is connected to a digital output.

Binding setup

96 digital points (BOOL) and 48 analog points (DINT) are transferred to another RTU32 every 1sec. The points are internal memory points, and are all changed in the same STRATON cycle.

Result:

The typical STRATON scantime is 4 to 7ms.

If data transfer are stopped (change of binding variables are stopped), STRATON scan time is typical 4ms. I.e. when variables are changed additional time (3ms) is added to generate the events and communication.

Example 4; STRATON application program with EN/IEC60870-5-101 communication

Small test configuration:

One EN/IEC60870-5-101 Slave unbalanced mode on COM2 9600 baud.

STRATON scan time is measured on the following application.

8 single point inputs (single point informations)

8 pulse outputs (single commands)

Result:

The typical STRATON scan time is 2 to 3ms.

Large test configuration:

One EN/IEC60870-5-101 Slave unbalanced mode on COM2 9600 baud.

STRATON scan time is measured on the following applications.

96 single point inputs (single point informations)

96 pulse outputs (single commands)

Result:

The typical STRATON scan time is 4 to 5ms.

Small test configuration:

One EN/IEC60870-5-104 Server.

STRATON scan time is measured on the following application.

8 single point inputs (single point informations)

8 pulse outputs (single commands)

Result:

The typical STRATON scan time is 2 to 3ms.

Large test configuration:

One EN/IEC60870-104 Server.

STRATON scan time is measured on the following applications.

96 single point inputs (single point informations)

96 pulse outputs (single commands)

Result:

The typical STRATON scan time is 4 to 5ms.

Comments to the example and measurements

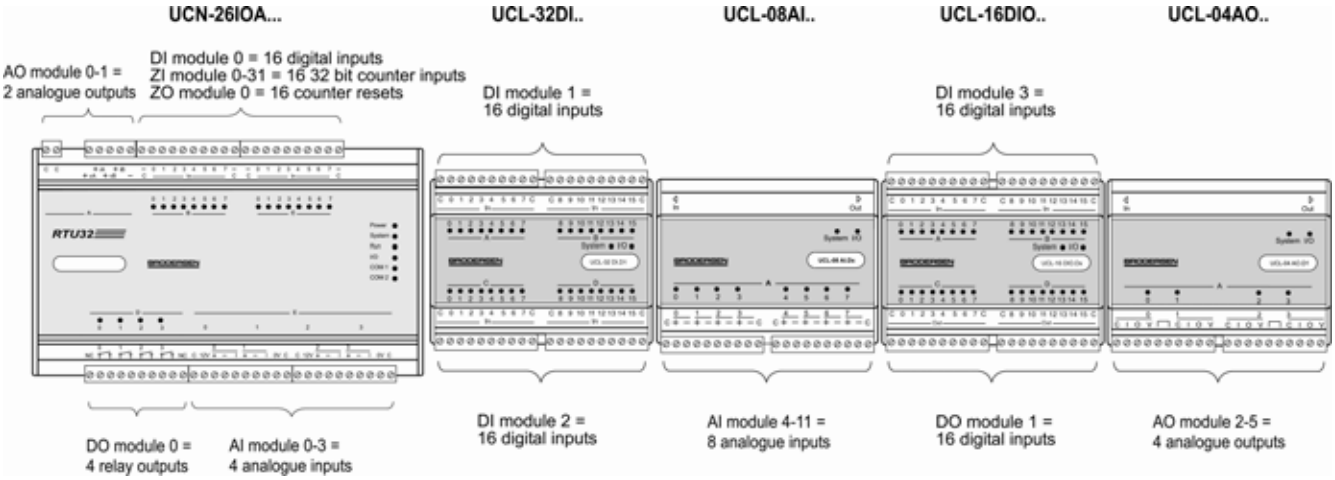
There are small differences in STRATON scan time whenever EN/IEC60870-5-101 or EN/IEC60870-5-104 communication is used. The ModbusRTU slave test showed also that no noticeable time was added to the STRATON cycle, due serial communication. It could be expected to be the same case for /IEC60870-5-101 and EN/IEC60870-5-104 communication.

WinCE remote performance monitor show the different communication types tested only contribute with a few percent (less than 5) of the CPU load. If many COM ports are added to the system, the load will increase, but should not be of general concern.

High load ping test at the Ethernet port show impact on the STRATON cycle time, and the large STRATON binding application test, show an increase in cycle time when all data are exchanged. I.e. the use of binding protocol to many RTUs, where lots of data are transferred at high rate, should be considered carefully if fast timing is an issue of concern.

Appendix 2

RTU32 I/O addressing example 1



Appendix 3

RTU32 I/O addressing example 2