STRATON SNMP Agent Driver

for

RTU32

V. 1.06 / Mar 2006 / Doc 40225

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

1 Introduction

Before you begin please note that:

The SNMP Agent driver for RTU32 system is made for alarm, monitoring and control purposes in network systems. The STRATON programming and setup environment offer the user full flexibility in how and what alarm and control features that are required in the application.

The SNMP Agent driver is an optional driver for the RTU32. It is only included if you have ordered the RTU32 with this option. The STRATON Workbench which is the programming and configuration tool, do need special libraries which. They are included in the Driver files.

The RTU32 do as default also include a standard SNMP Master Agent for basic operating information's as controller load and network traffic statistic etc. For documentation and description of this please see the applicable homepage for Microsoft Embedded software WinCE 5.0.

The SNMP Driver guide include descriptions related to the SNMP Driver it selves. It is expected that you already have basic knowledge of setting up the RTU32 and programming experience in the STRATON Workbench.

RTU32 SNMP Driver feature list

Basic features

The RTU32 is running MS WinCE version 5.0, which always include a basic SNMP Master Agent plus a basic Extension Agent. The basic Extension Agent is part of the Microsoft environment for reporting network information's etc.

The RTU32 SNMP Driver for STRATON is added as an additional Extension Agent in the RTU32. The purpose of this Agent is to use the RTU32 STRATON application program for reporting alarm Traps and reading/writing acquisition data of any variable inside STRATON. Variables could be direct I/Os or any internal variable generated in STRATON.

The advantages of linking together the SNMP Agent with the STRATON is that it offers full flexibility in setting up any thinkable configuration for alarming and monitoring of equipment, environmental data and asset management.

The SNMP Agent comply to the SNMPv2C standard.

Supported SNMP Agent Driver features

The SNMP Agent driver for STRATON supports the SNMP functions;

Traps. Alarms send as events from the managed device – RTU32 - to the SNMP Manager.

Get / GetNext. RTU32 reply to Get/GetNext requests from the SNMP Manager.

Set. Execute Set commands send from the SNMP Manager.

In the SNMP Agent Driver in STRATON each function type are available as number of predefined Function Blocks. In the STRATON Workbench you can create your specific application program according to your requirements. The application program can take data from physical digital or analogue I/Os, data from other interface protocols and internal defined variables and link them to any SNMP Agent data function as required.

As an example you can with the SNMP functions in STRATON can report and control any kind of analogue data like temperature, humidity, voltage and current levels, alarm setpoints etc.

Alarms can be configured to be according to your requirements. I.e. use any limit alarm levels, grouping and prioritizing alarms etc.

In addition the RTU32 with SNMP Agent Driver do offer you powerful local controller functions. These can be linked to alarms related to the SNMP requirements, but also be totally independent if required.

2 Installation

General

The SNMP Driver for RTU32 is optional and is not implemented on standard RTU32. The Driver can be available in 2 ways:

Preinstalled at RTU32

The RTU32 is ordered with this option. All necessary files and settings are pre-installed in the RTU32.

Ordered as an additional driver to an RTU32

The SNMP Driver can be ordered as a stand-alone driver to the RTU32. In this case you receive a Flash disc with a new WinCE image for your RTU32 and a disc with the necessary library files for STRATON. The RTU32 will then work just like an RTU32 with the driver pre-installed SNMP Agent driver. The library files has to be placed on your workstation running the STRATON Workbench.

STRATON Hardware library

Om the disc included with the SNMP Driver you will find the SNMP hardware library for 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 PC.

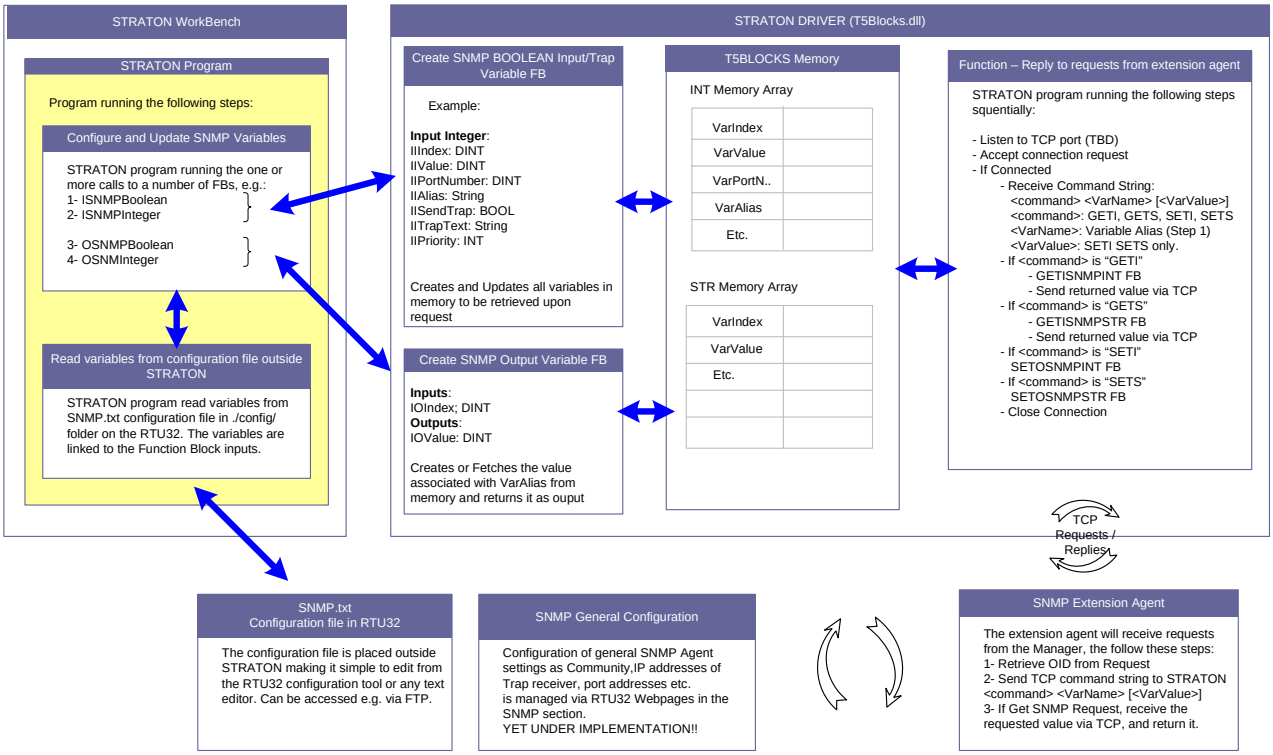
In the STRATON Workbench you will now have functions for SNMP and INIFILES. The SNMP functions id directly the functions to handle SNMP Agent data. The INIFILES functions are for reading variables from a configuration file (.txt file) into your application program. The idea is to provide the possibility to store variables as settings, alarm text strings, device data etc. out of the STRATON application program.

3 SNMP Agent driver details

General

The driver for SNMP is basically include 2 main parts; the Extension Agent in WinCE which handle the SNMP driver communication to the SNMP Manager on the network and the STRATON driver part which link together the STRATON variables to the SNMP functions running in the operating system (WinCE). See the block diagram figure 1.

Figure 1



The RTU32 is id' in managed networks with its own branch name/number which for Brodersen Controls A/S is 24122.

SNMP Agent basic functions

The SNMP Agent Driver support the following SNMP functions:

Traps

Get Request / GetNext Request

Set Request

These functions are handled by the SNMP driver part in the operating system. I.e. the driver reply to SNMP Request type Get when the Manager is sending a Get request.

The data provided for these request is made available in a memory area by the STRATON SNMP Agent driver part. You defined in the STRATON application program a number of variables, which is placed in a memory area ready for passing to the manager if requested. The variables for the SNMP data include values and strings according to the STRATON function block section. Trap data alone do not require a MIB as they have direct description strings included.

In the case that the variable in STRATON is defined as a Trap, the alarm Trap will immediately be handled by the SNMP driver part in WinCE and send to the SNMP Manager with the defined IP address / addresses.

MIB and OIDs

General

The MIB file for an RTU32 SNMP Driver application is used for defining the RTU32 SNMP variables for a SNMP Manager. The MIB file is entered in the SNMP Manager software to translate SNMP data send as pure numeric data (OIDs) from the Agent to understandable text strings used in the SNMP manager. The OID (reply from the Agent driver) is defined by the RTU32 branch number plus some numbers that defined the actual type of data, i.e. an alarm initiated by an activated digital input.

Example of an OID: <enterprices>.24122.1.2.3.4.5.0

Where the:

24122	=	Brodersen Controls A/S branch id
.1	=	RTU32 product id
.2	=	SNMP data types: 1= device data, 2=Input variables, 3=output variables
.3	=	Define type of data; 1=Boolean, 2=Integer, 3=String if Input/Output. If DeviceData the first DeviceData is set to 1 and automatically incremented with 1 for each DeviceData defined for the device.
.4	=	Define the Index value set in the Function block.
.5	=	Define the actual variable in the Function block starting with 1 and counting upwards. I.e. in the ISNMP function block 1=XValue, 2=XAlias, 3=TrapText, 4=Xpriority, 5= port id.
.0	=	Every OID is ended with .0 as all OIDs are variables.

OID structure for RTU32

OID structure is shown below with limited data types:

DeviceData <enterprices>.24122.1.1

- DeviceData1 <enterprices>.24122.1.1.1.0
- DeviceData2 <enterprices>.24122.1.1.2.0
- DeviceData2 <enterprices>.24122.1.1.3.0
- ….
- DeviceDatax <enterprices>.24122.1.1.x.0

RTU32InputVariable <enterprices>.24122.1.2

- RTU32BoolInputVariable 1: <enterprices>.24122.1.2.1.1.0
  - BIVar1Value <enterprices>.24122.1.2.1.1.1.0
  - BIVar1Alias <enterprices>.24122.1.2.1.1.2.0
  - BIVar1TrapText <enterprices>.24122.1.2.1.1.3.0
  - BIVar1Priority <enterprices>.24122.1.2.1.1.4.0
  - BIVar1PortNumber <enterprices>.24122.1.2.1.1.5.0
- RTU32BoolInputVariable 2: <enterprices>.24122.1.2.1.2.0
  - BIVar2Value <enterprices>.24122.1.2.1.2.1.0
  - BIVar2Alias <enterprices>.24122.1.2.1.2.2.0
  - BIVar2TrapText <enterprices>.24122.1.2.1.2.3.0
  - BIVar2Priority <enterprices>.24122.1.2.1.2.4.0
  - BIVar2PortNumber <enterprices>.24122.1.2.1.2.5.0

- RTU32IntInputVariable 1: <enterprices>.24122.1.2.2.1.0
  - IIVar1Value <enterprices>.24122.1.2.2.1.1.0
  - IIVar1Alias <enterprices>.24122.1.2.2.1.2.0
  - IIVar1TrapText <enterprices>.24122.1.2.2.1.3.0
  - IIVar1Priority <enterprices>.24122.1.2.2.1.4.0
  - IIVar1PortNumber <enterprices>.24122.1.2.2.1.5.0
- RTU32IntInputVariable 2: <enterprices>.24122.1.2.2.2.0
  - IIVar2Value <enterprices>.24122.1.2.2.2.1.0
  - IIVar2Alias <enterprices>.24122.1.2.2.2.2.0
  - IIVar2TrapText <enterprices>.24122.1.2.2.2.3.0
  - IIVar2Priority <enterprices>.24122.1.2.2.2.4.0
  - IIVar2PortNumber <enterprices>.24122.1.2.2.2.5.0

- RTU32StrInputVariable 1: <enterprices>.24122.1.2.3.1.0
  - BIVar2Value <enterprices>.24122.1.2.3.1.1.0
  - BIVar2Alias <enterprices>.24122.1.2.3.1.2.0
  - BIVar2TrapText <enterprices>.24122.1.2.3.1.3.0

BIVar2Priority <enterprices>.24122.1.2.3.1.4.0
BIVar2PortNumber <enterprices>.24122.1.2.3.1.5.0

RTU32StrInputVariable 2: <enterprices>.24122.1.2.3.2.0
BIVar2Value <enterprices>.24122.1.2.3.2.1.0
BIVar2Alias <enterprices>.24122.1.2.3.2.2.0
BIVar2TrapText <enterprices>.24122.1.2.3.2.3.0
BIVar2Priority <enterprices>.24122.1.2.3.2.4.0
BIVar2PortNumber <enterprices>.24122.1.2.3.2.5.0

RTU32St… etc.

RTU32OutputVariable: <enterprices>.24122.1.3
RTU32BoolOutputVariable 1: <enterprices>.24122.1.3.1.1.0
RTU32BoolOutputVariable 2: <enterprices>.24122.1.3.1.2.0
RTU32BoolOutputVariable x: <enterprices>.24122.1.3.1.x.0

RTU32IntOutputVariable 1: <enterprices>.24122.1.3.2.1.0
RTU32IntOutputVariable 2: <enterprices>.24122.1.3.2.2.0
RTU32IntOutputVariable x: <enterprices>.24122.1.3.2.x.0

RTU32StrOutputVariable 1: <enterprices>.24122.1.3.3.1.0
RTU32StrOutputVariable 2: <enterprices>.24122.1.3.3.2.0
RTU32StrOutputVariable x: <enterprices>.24122.1.3.3.x.0

The MIB file must be edited according to this structure. A number of MIB files examples are provided. The MIB files examples include a number of variables and predefined comments, which should guide you in editing the MIB file.

4 General settings

SNMP Agent Driver basic settings

The WebServer contain webpages for configuring the basic SNMP settings for the SNMP Driver. This includes configuration of up to 4 community.

NOTE: The LAN2 is the default port used for SNMP communication. And LAN2 must be set to DHCP to avoid cold start traps to be routed to this port.

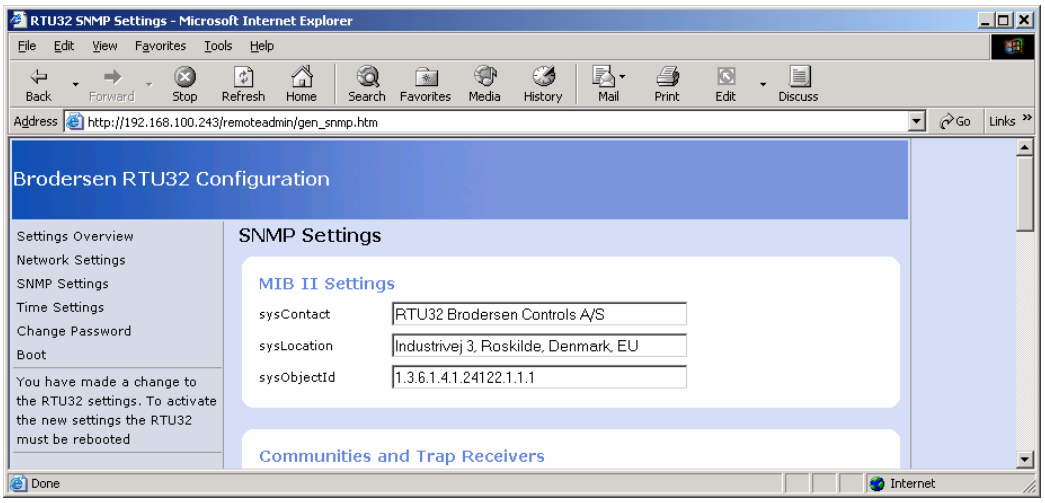
SNMP settings

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 configuration parameters for the SNMP Agent Driver are;

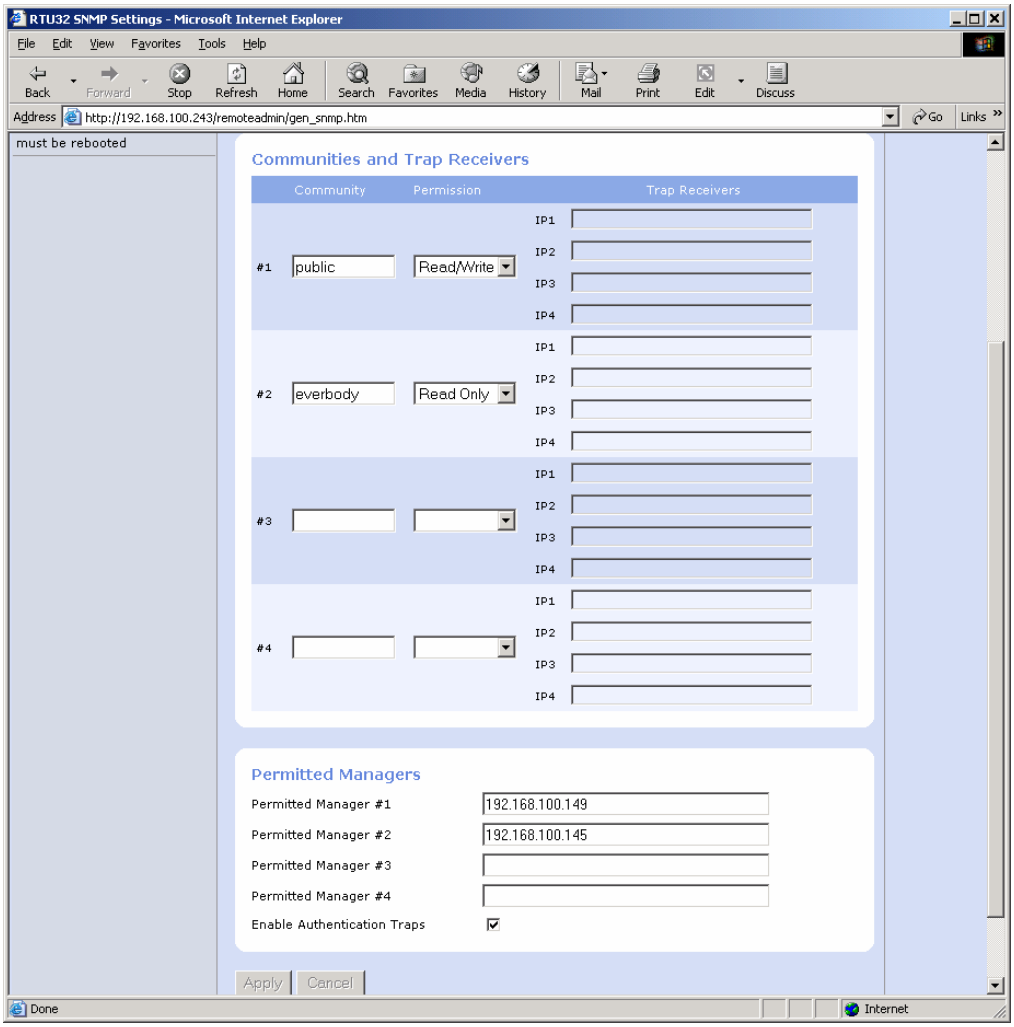
MIB II Settings

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.

Setting up variables

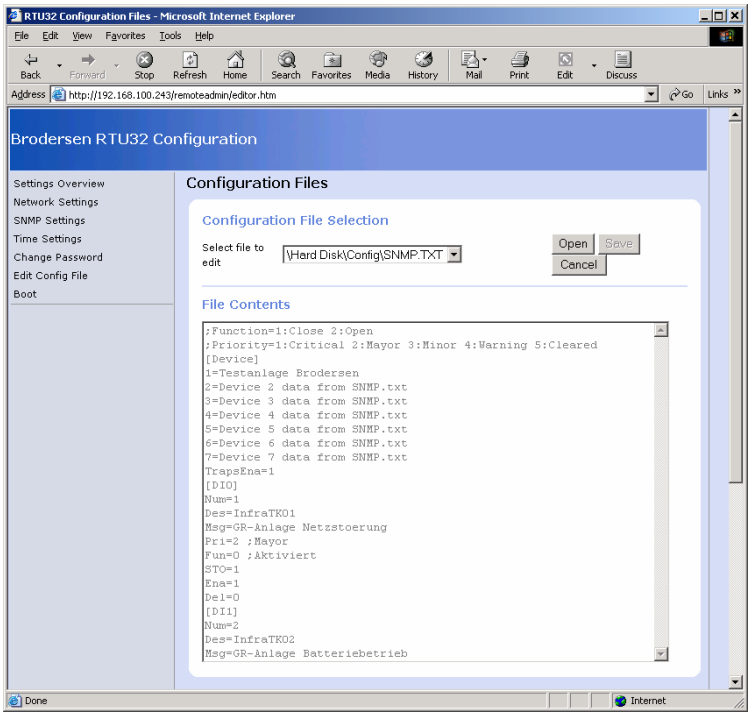
You can define your variables directly to the SNMP FunctionBlocks or if required use variable data read from a configuration file.

A configuration file can be used in the SNMP STRATON application program to setup SNMP parameters and variables without changing the STRATON program itself.

The STRATON functions in INIFILES can be used for reading variable values and strings from the configuration file. An example program used the configuration file SNMP.txt. It is placed in /HARDDISK/CONFIG/ folder in the RTU32.

Remote configuration of SNMP Driver variables

A simple text editor is available on the web server under the menu Edit Config File. You can remotely access the Webpages and use the available text editor to edit the configuration files. You must select the SNMP.txt and open it. After making your adjustments, you just select save to store the changes in the RTU32.



Optional you can use the **RTU32Upgrade** software, which has a function to edit configuration files. It gives you possibility for remotely access to upload the file, edit it and download the SNMP.txt configuration file. Please ask for example application programs where this option is used.

NOTE: The configuration file is application specific and do correspond with the STRATON Application programming. You can either adapt the default file settings or make your own. The default SNMP.txt do comply with the SNMP STRATON application demo program.

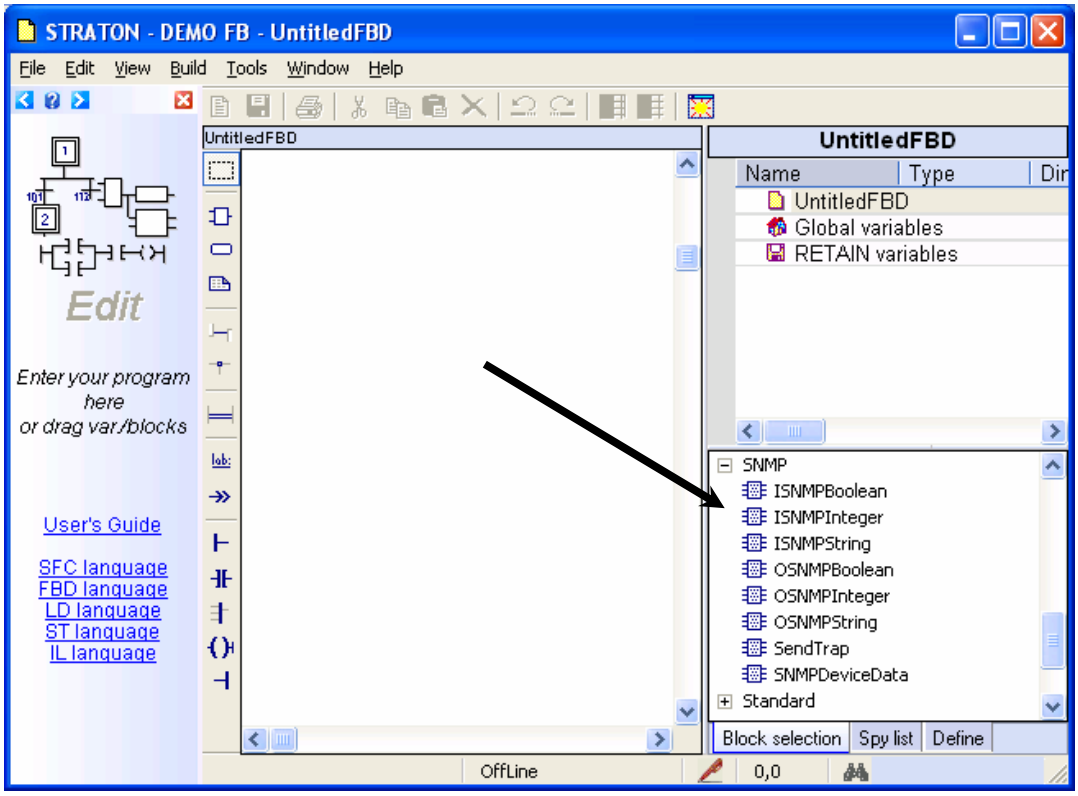
5 STRATON Function blocks

STRATON SNMP FunctionBlocks

By means of Brodersen RTU32 SNMP libraries added to STRATON WorkBench, it is possible for the application programmer to link any I/Os, internal or global variables to the SNMP Driver. The variables can be Boolean, integers or strings of up to 255 characters.

The STRATON program takes care of linking the I/Os to the physical inputs/outputs. This way special requirements for set point alarms on analogue inputs and grouping alarms can be handled. Also special variables can be assigned to a trap alarm.

STRATON Function Blocks for inputs



There are 3 types of function blocks for inputs;

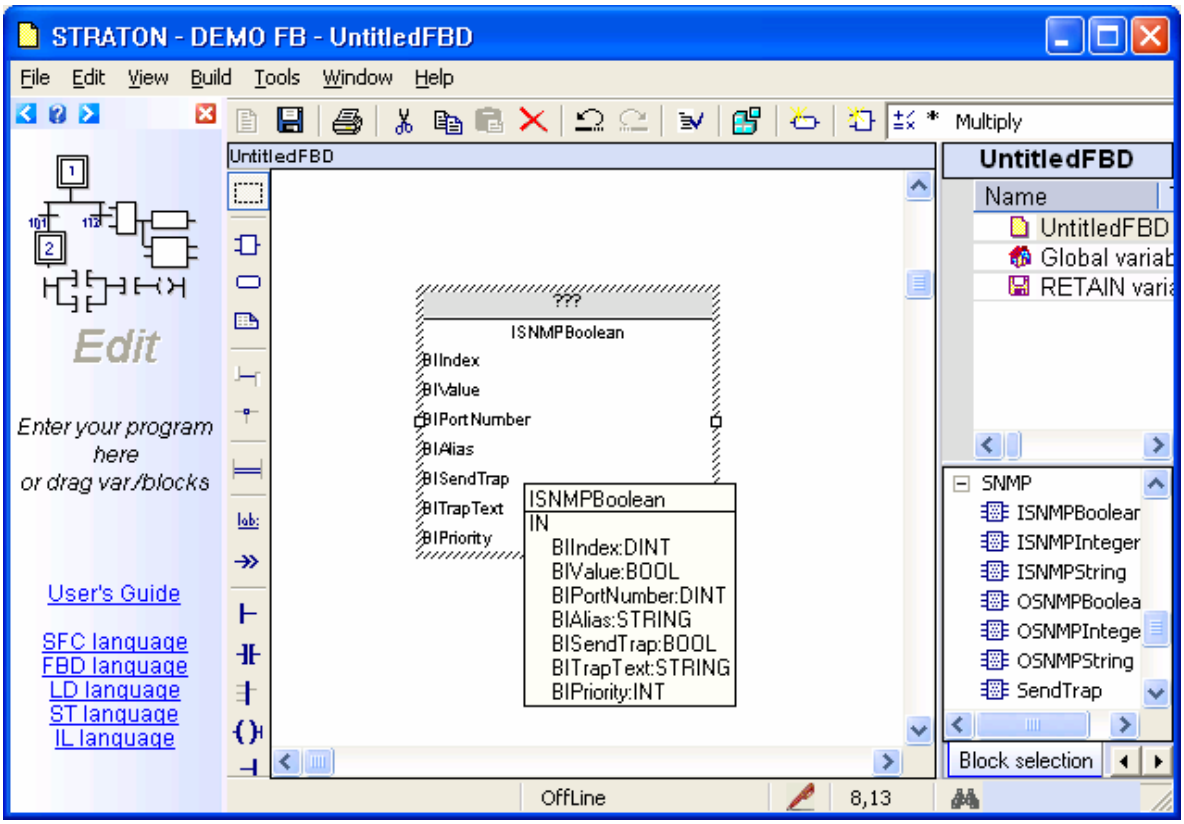
ISNMPBoolean for reporting Boolean inputs

ISNMPInteger for reporting Integer inputs

ISNMPString for reporting Strings

The ISNMPxxx functions provide data for SNMP Get and GetNext requests. In addition they are also used for generating Traps.

Description of Function Block input / output



BIIndex: Id of Boolean information. Each type must have a unique id in the OID.

BIValue: Value of the input (Boolean).

BIPortNo: No of port (DINT).

BIAlias: Description of the port (String; ASCII and max. 255 characters).

BISendTrap: Input activating sending as Trap (Boolean).

BITrapText: Descriptive text send with the Trap (String; ASCII and max. 255 characters).

BIPriority: Priority of Trap – set as value e.g. as in Priority section (INT).

For the Integer and String input type, the value is replaced by the data type and the PortNumber.

Function Blocks for outputs

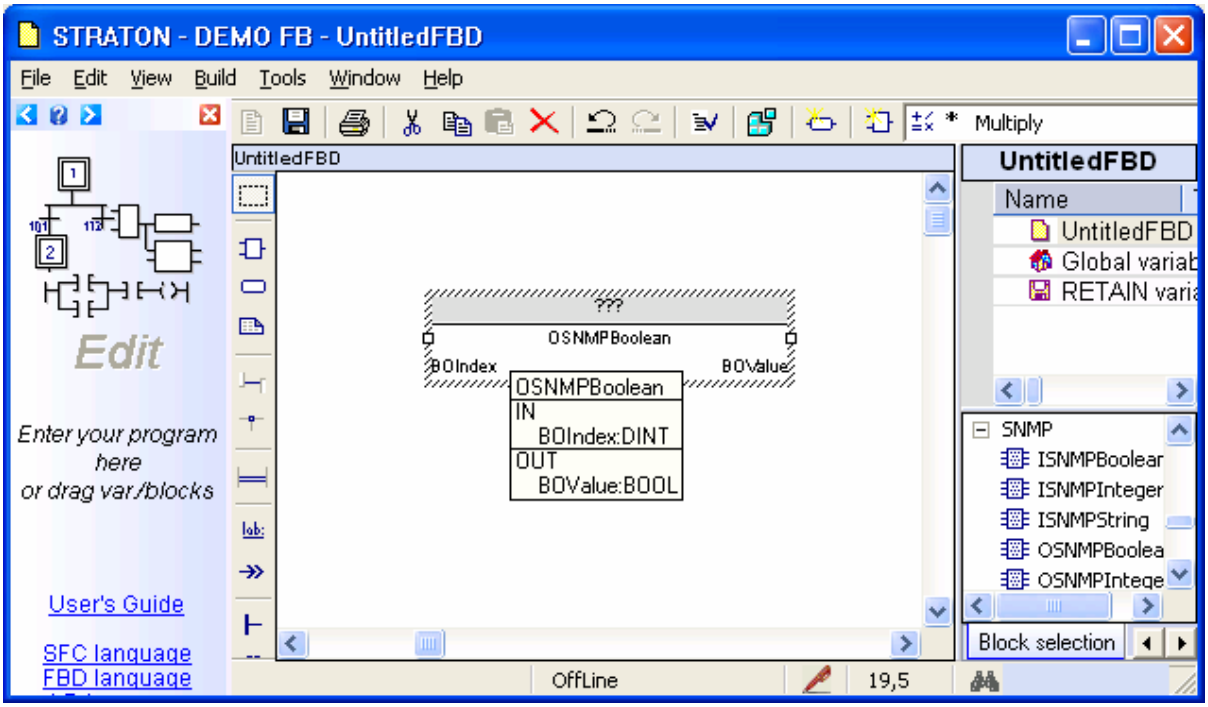
The is 3 types of function blocks for output;

OSNMPBoolean for setting Boolean outputs

OSNMPInteger for setting Integer outputs

OSNMPString for setting Strings

Description of Function Block input / output



BOIndex: Id of Boolean information. Each type must have a unique id in the OID (DINT).

BOValue: Value of the output (Boolean).

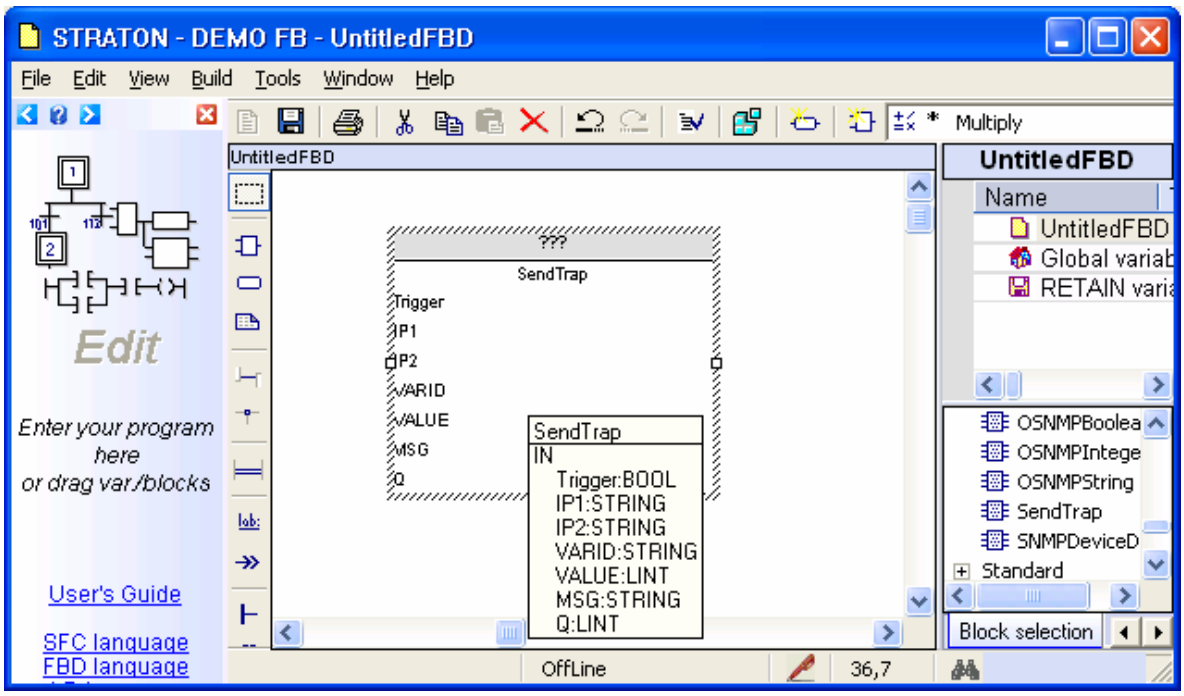
For the Integer and String type, the value is just replaced by the data type.

Function Blocks for Traps

A Function Block is provided for sending traps only. It is different from the ISNMPxxx functions in regards to:

- You can define 1 or 2 IP addresses directly as a variable on the function. It does not use the IP addresses setup in registry via the Webpage.
- You are not able to use the Get data and GetNext data to read these actual values.

Description of Function Block input / output



Trigger: Input activating sending Trap (Boolean).

IP1: IP address of the first Manager to send a Trap (String).

IP2: IP address of the second Manager to send a Trap (String).

VARID: Description of alarm type (String -ASCII and max. 255 characters).

VALUE: Value which must be reported with Trap (LINT).

MSG: Alarm text of the specific condition (String -ASCII and max. 255 characters).

Q: Priority of Trap – set as value e.g. as in Priority section (LINT).

Note that all inputs and output for the function blocks always must be connected. If not used constants has to be applied.

SNMP Severity settings for inputs

Each input type include assignment of for example five possible priorities. The priority options could be:

Critical – set the value 1

Major – set the value 2

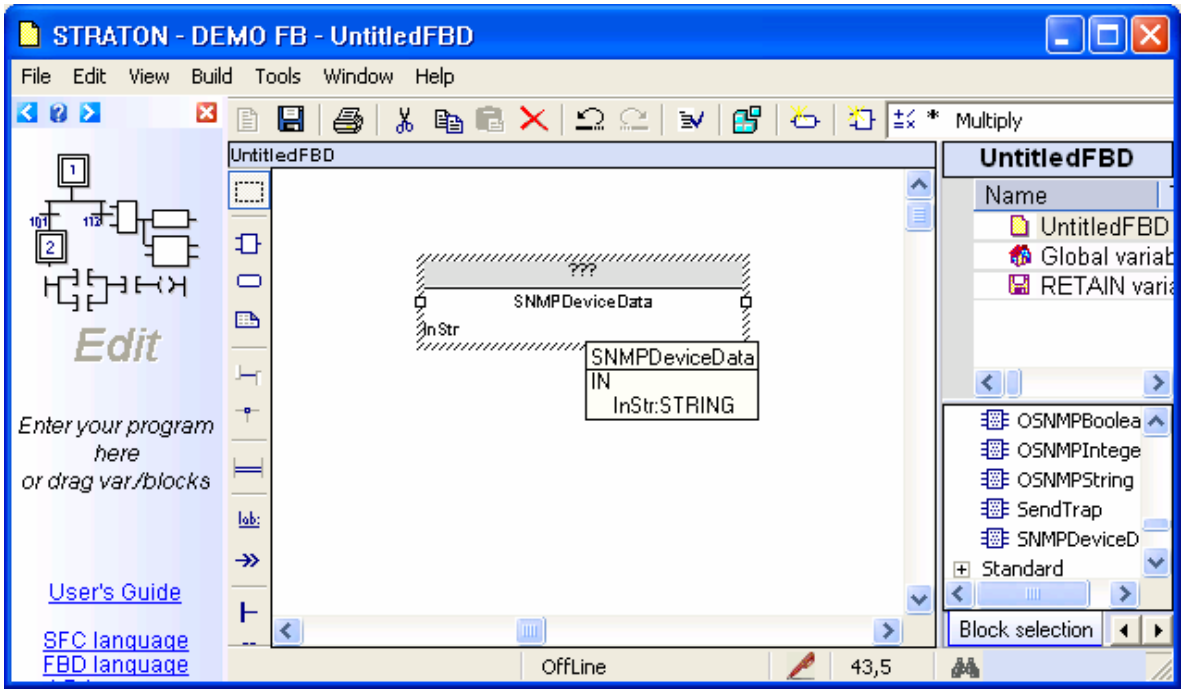
Minor – set the value 3

Warning – set the value 4

Cleared – set the value 5

The setting can be defined by the user – each value mean what the user use in the SNMP Manager application.

Function Blocks for Device data



The function block for defining device data for the RTU32 is used for reporting device information. Is send as String data. You can add as many as you like and they will in the SNMP Management software be listed as they are put into your application program.

InStr: String input describing Device data. In case you want to report more than one device data information, you just add more Device data function blocks. They will be reported in the sequence they are defined in your application program.

GenericTraps

Some GenericTrap is generated from the RTU32. This include:

ColdStart: When the Application program is started the RTU32 send a ColdStartTrap. Note; report LAN1 IP always if LAN2 is set to DHCP.

AutenticationFailure: When unauthorized persons try to login to the RTU32 via e.g. FTP or Webbrowser and fails, an AuthenticationTrap will report the event and the Trap will include the IP address of the device causing the event. Is setup on the web pages SNMP general settings.

6 Optional configuration functions

Configure or re-configure the RTU32 with a single file

The RTU32 offers some functions to read, write and edit Registry files in the WinCE OS. With these functions it is possible to send files via FTP to the RTU32 which can make a complete configuration of the RTU32. After placing the basic configuration file you just reboot the RTU32 and during boot it will run and execute the setting defined in the configuration file.

NOTE: This configuration option details is not documented in this Manual. And basically it is only recommended to be used by skilled software programmers.