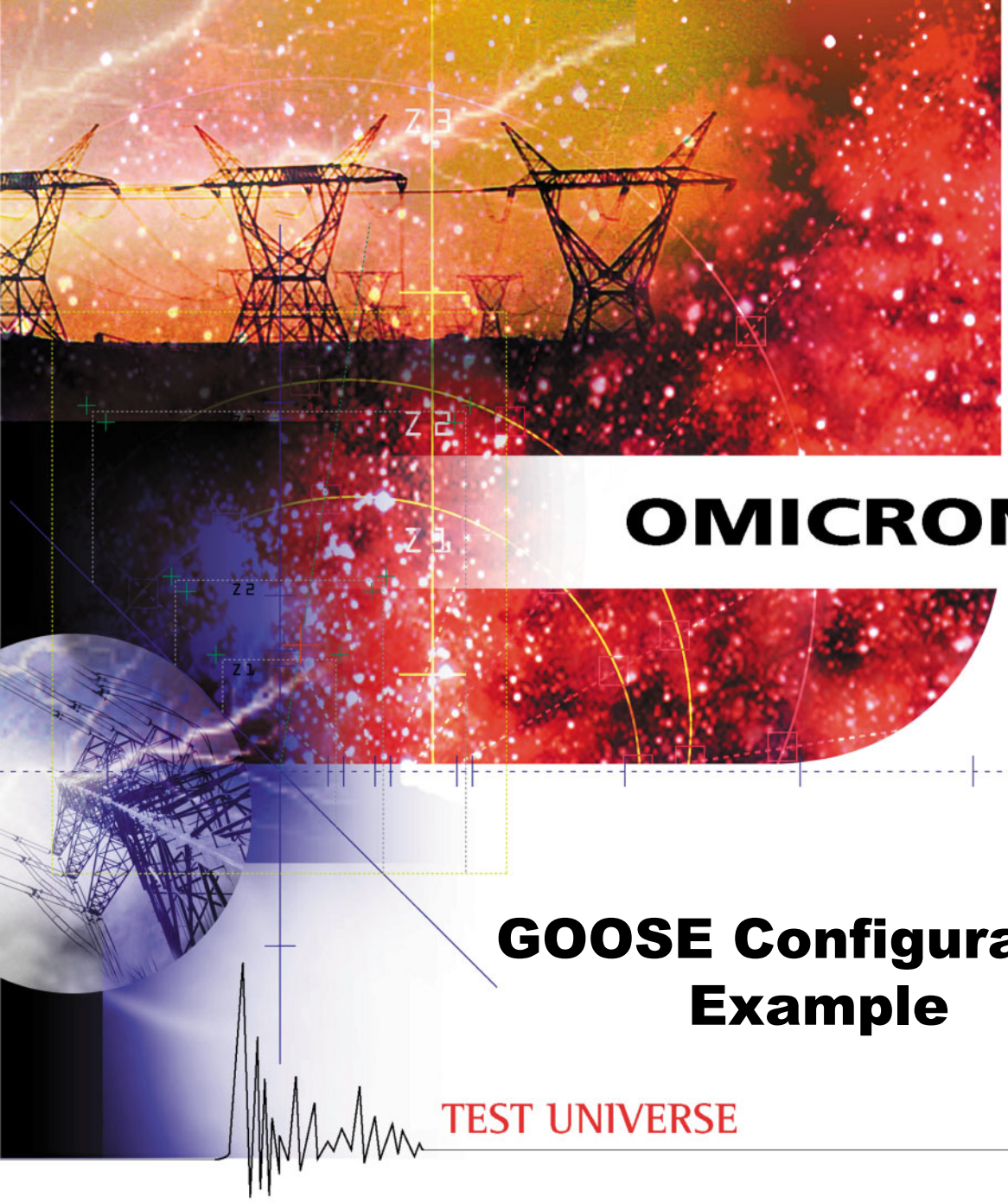

OMICRON

GOOSE Configuration Example

TEST UNIVERSE

Manual Version: GOOSE-CMC.AE.5 - Year 2010

Testing an IEC 61850 Compliant Relay with a *CMC* Test Set and the *GOOSE Configuration* Module

Introduction

This example focuses on the configuration of the GOOSE (IEC 61850 GOOSE) features of a CMC Test Set. The details of protection testing are not covered. Thanks to OMICRON's smart implementation of the features, the established methods for protection testing are not affected by the new signaling mechanism.

The detailed explanation of the internal structure of a relay modeled according to IEC 61850 and the related services is also beyond the scope of this document. In the following, it is assumed that the reader is familiar with the basic principles of IEC 61850.

Terms: IEC 61850, GSE, and GOOSE

IEC 61850, the international standard for communication networks and systems in substations, defines (among many other items) mechanisms for real-time messaging over Ethernet networks.
The general term for this is GSE (Generic Substation Event). There are two specific forms of this, the GSSE (Generic Substation State Event), and the GOOSE (Generic Object Oriented State Event).

The GOOSE messages have configurable contents (the GOOSE Dataset) and can contain any valid data type defined in IEC 61850 as long it fits into a single Ethernet packet.

The *GOOSE Configuration* module makes this task as easy as possible for the user, and provides automatic configuration during testing and proper documentation of the settings made.

The UCA 2.0 specification (Utility Communication Architecture) can be regarded as a predecessor of IEC 61850 in certain aspects. It contained already a real-time messaging concept, also called GOOSE.
This "old UCA GOOSE" is equivalent to the GSSE defined in IEC 61850.

The expression "GOOSE" has been taken over by IEC 61850 and used for something really object-oriented, the former "UCA GOOSE" has become defined under the new term "GSSE".

The GSSE messages are one-dimensional arrays of (double-bit) status indicators.

The Test Set

For testing with binary status signaling via GOOSE, either a *CMC 256* with *NET-1 Option*, a *CMC 256plus*, a *CMC 353*, a *CMC 356* or a *CMC 850* is required.
In the remainder of this document, this specific type of equipment is simply referred as the "Test Set".
It provides the following features:

- Capturing GOOSEs from the network and reacting on specific status information in the messages,
- Simulating IEDs to send out GOOSEs,
- Configuration software module,
  which is used to set up the subscriptions for receiving GOOSEs and to set up the simulation of IEDs for sending GOOSEs. This module can be embedded in an *OMICRON Control Center* test plan to reconfigure the test set during a test of a multifunctional relay.

OMICRON has integrated the GOOSE features into the test set in a way that the status information in the messages on the network are mapped to the binary inputs and outputs of the test set. The same ten binary inputs and eight binary outputs, which are used by most of OMICRON's test modules, are available for the GOOSE mapping.

The mapping is performed via the *GOOSE Configuration* module and is kept completely outside of the other test modules. The test modules do not even know if a binary input is operated by a hard-wired contact or a status bit in a GOOSE. In the same way, the test modules do not know if setting a binary output will cause a status indication in the messages of a simulated IED.

This way, all OMICRON test modules are immediately available for testing IEC 61850 compliant IEDs. The method of using the protection testing functions of the software remains completely unchanged and does not need to be further explained in this example.

About the Internet Protocol

The Internet Protocol (IP) is not used for the real time messaging via GOOSE.

However, it is used for controlling the test set from the PC. The related settings are made in the test set *Association and Configuration* utility which can be accessed from the OMICRON *Start Page*. Further details can be found in the related help and documentation.

The IP may be also used for setting the relay. In this given case, the relay setting program can access the relay via the Ethernet using the IP.

Most relays use a serial interface to set an IP address. The devices only work with fixed (static) IP addresses. The IP settings of the relays must always be set manually to match the network configuration that the relays are connected to.

Protection Testing with Subscribed GOOSEs

The general trip and start signals are subscribed and used for testing the protection function.
The signal flow is outlined in the figure below.

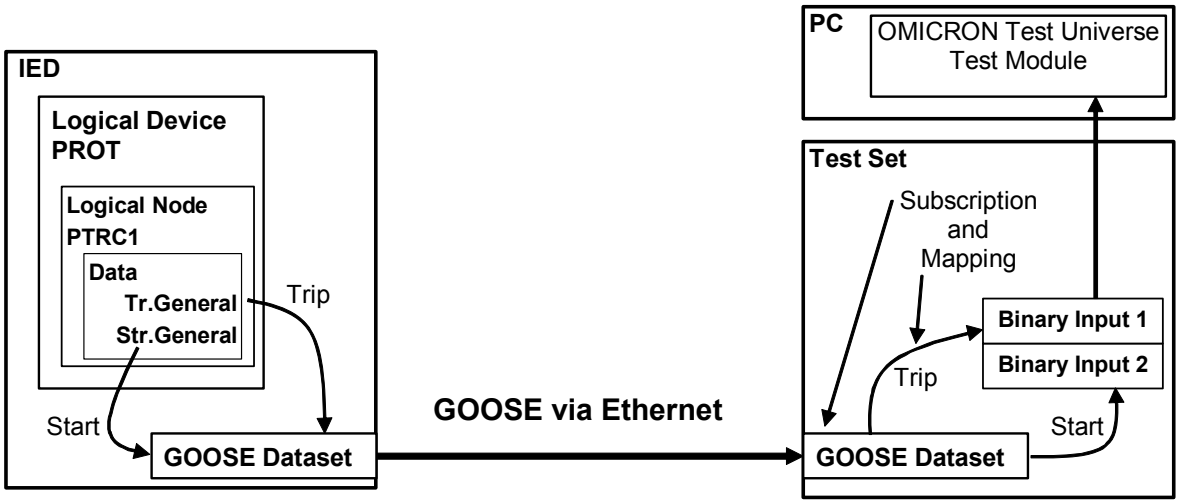

Fig. 1 GOOSE mappings in the relay and test set

The IED contains (among other logical devices) a logical device "PROT", which serves as a container for all the protection-related sub functions (logical nodes). One of these logical nodes is "PTRC1" of the type PTRC (Protection Trip Conditioning), which contains data such as the general trip and start information.

These data are mapped in the relay into the dataset which is sent out with the published GOOSE messages.

The test set subscribes these GOOSE messages and maps the data items from the received dataset to binary inputs of the test set. This is accomplished by the *GOOSE Configuration* module. All further processing in the protection testing SW (Test Modules of the OMICRON Test Universe) refers to these mapped binary inputs.

How to Set Up a GOOSE Configuration

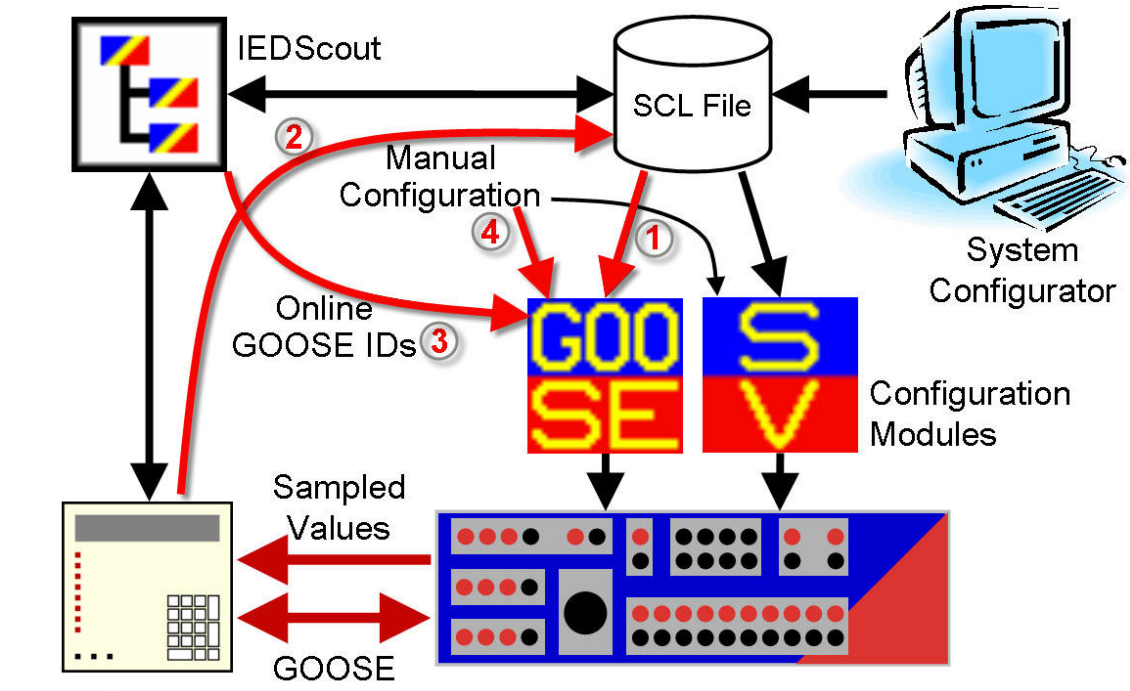

Fig. 2 Workflow to configure the test set

Figure 2 indicates the workflow of the four basic methods to configure the test set:

- Using the SCL file from the engineering process (system configurator)
- Using an SCL file created from the IED's self description
- Online GOOSE sniffing
- Manual configuration of the subscription

The four methods are described in detail below.

(1) Using the SCL File from the Engineering Process

This method assumes that an SCL file is available. Typically, an SCL file is created from a system configuration tool during the engineering process. Another ways to obtain an SCL file is explained below (see method (2)).

(1a) Importing the SCL File

Start the *GOOSE Configuration* module and import the SCL-file by selecting File -> 'Import Configuration…'. The SCL-file is parsed and all contained GOOSE definitions are offered in a further dialog. The GOOSE can be individually selected for subscription, simulation, or both.

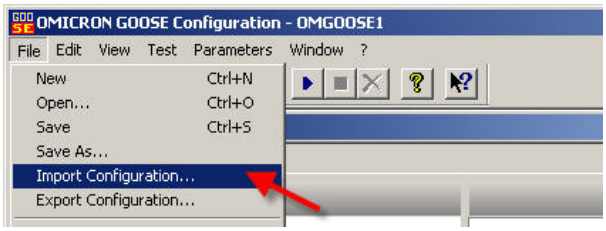

Fig. 3 Importing the SCL

This way, all configured GOOSEs from the SCL-file will be added in the *GOOSE Configuration* module without any typing.

(1b) Mapping the Data Attributes

After the SCL file has been imported the GOOSEs have to be mapped to the binary inputs of the Test Set. Therefore select the binary input to be mapped and move a data item from the received GOOSE by Drag & Drop to this input. Repeat this until all mappings have been established.

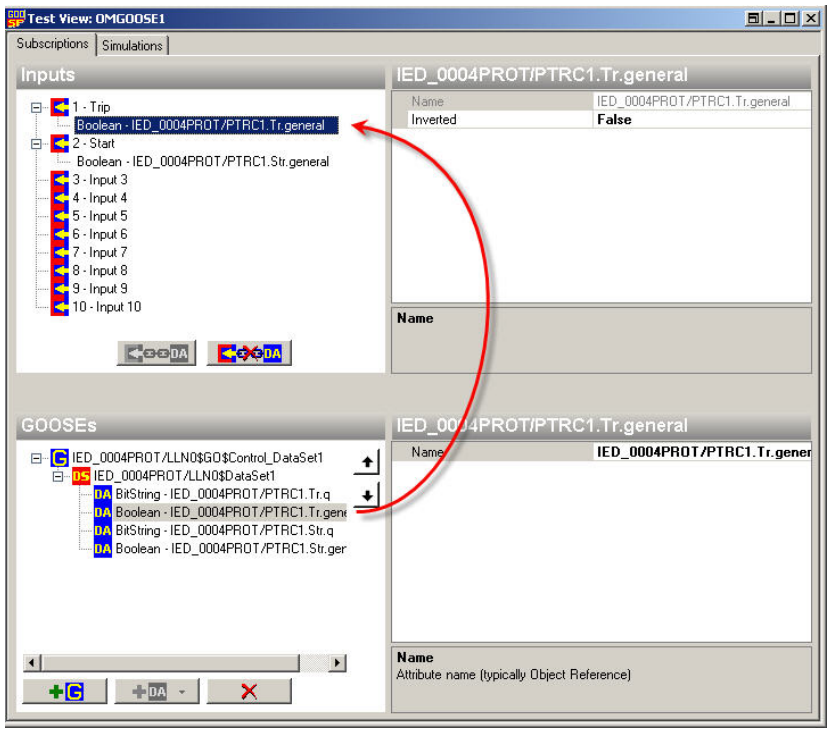

Fig. 4 Mapping data from the GOOSE to binary inputs of the test set

The test set provides the ten binary inputs and up to 360 virtual inputs for the mapping. The mapped data can come from up to 128 subscribed GOOSEs.

(1c) Applying the Configuration to the Test Set

By pressing the start  button, the GOOSE configuration is transferred to the test set.

The progress bar indicates the progress of the configuration process. The "Test State" indicates if the new configuration could be successfully applied.

(2) Creating an SCL File

This method is useful if no SCL file from the engineering process is available.

For creating a SCL file of the IED, *OMICRON IEDScout* is used.

First, the IED to be tested must be defined *IEDScout*. This is done by creating a new server in the configuration. In most cases, the IP address is sufficient to define the server.

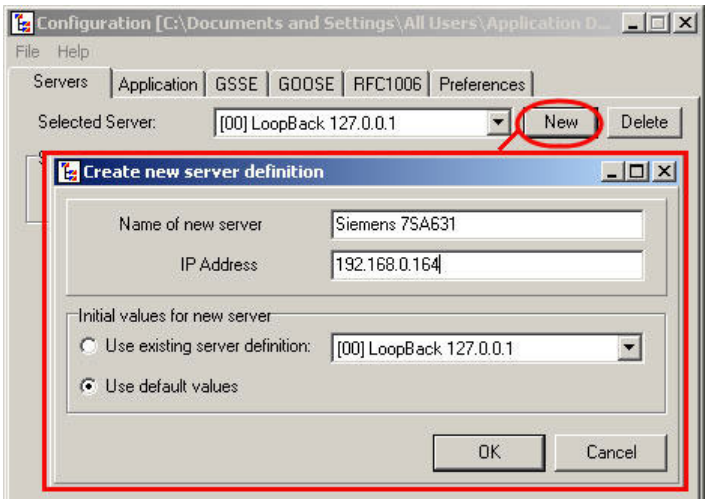

Fig. 5 Defining a new server

The next step is to discover the IED by selecting the 'Discover'-function in the main window of the *IEDScout* and to connect to the server.*

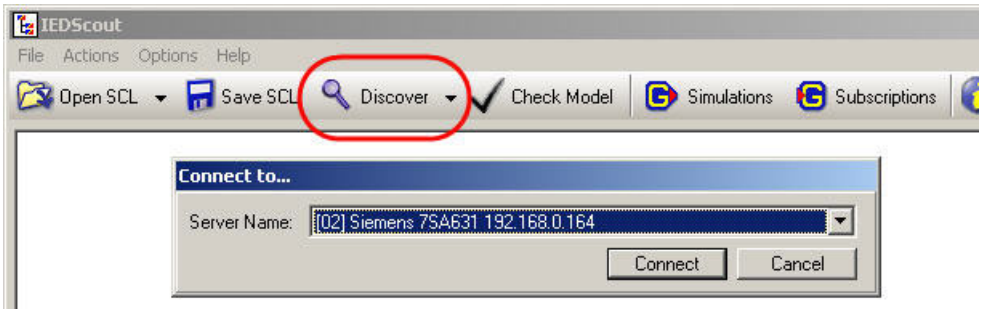

Fig. 6 Discovering the server

Now the *IEDScout* reads out the self description of the connected IED. These data can than be saved in SCL format by selecting File -> 'Save SCL…'.

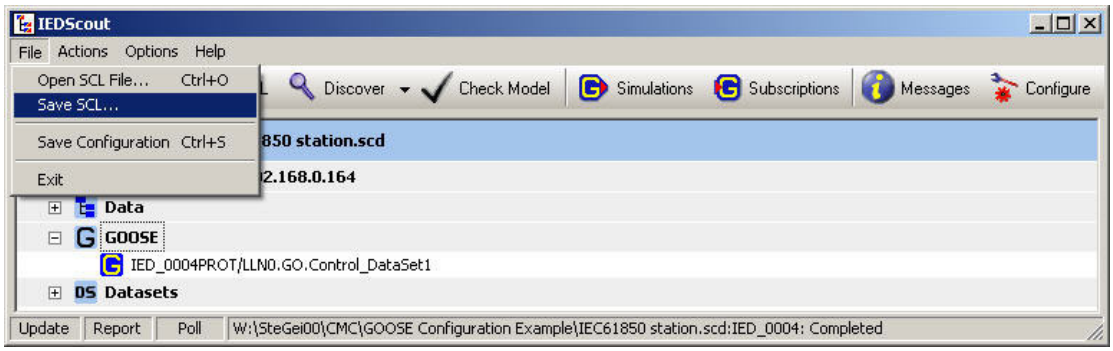

Fig. 7 Saving an SCL file

The SCL file obtained this way can then be used as described in method '(1) Using the SCL file'.

* For more detailed information on IED browsing and saving SCL files refer to the *IEDScout Example* file, which can be found in the *IEDScout* installation folder.

(3) Online GOOSE Sniffing

This method supplies information to the *GOOSE Configuration* module without using an SCL file. Again, IEDScout is used for this method.

Therefore open the *IEDScout* and the *GOOSE Configuration* module.

In the *IEDScout* window, open 'Subscriptions' and select the 'Sniffer' function.

A discovered GOOSE can be transferred to the *GOOSE Configuration* module by using Drag & Drop or Copy & Paste.

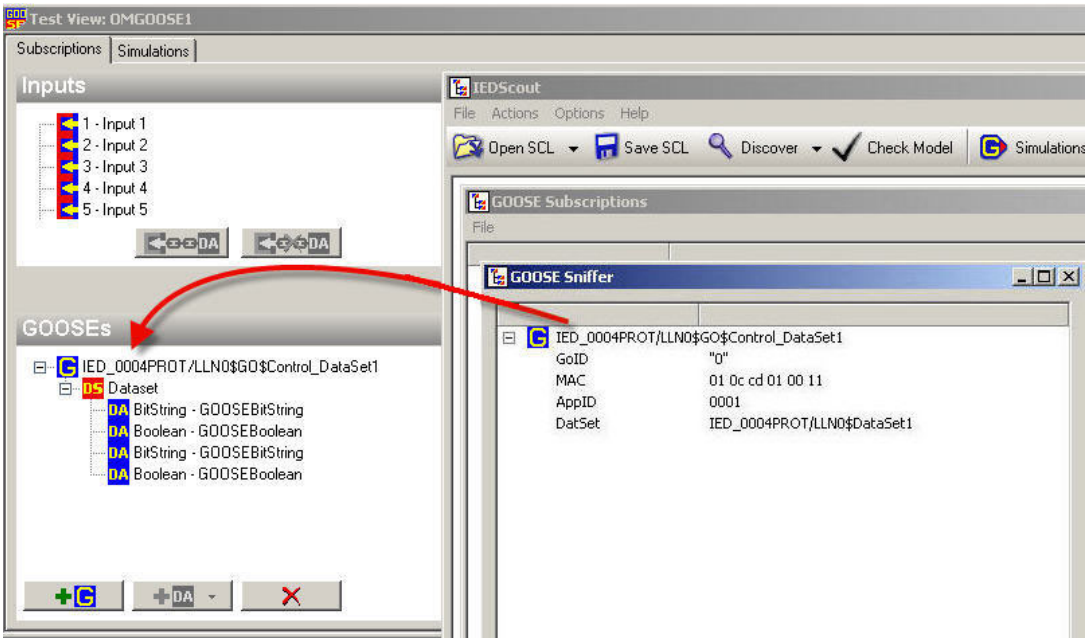

Fig. 8 Copying the GOOSE information into the GOOSE Configuration module

When transferring the GOOSE information from *IEDScout*, only the data type information is provided to the *GOOSE Configuration* module. The information about the actual meaning (the semantics) of the data (Start, Trip,…) is not available when using this method. For getting such information, see also "Determining the GOOSE Details" below.

The configuration of the test set can then be continued with '(1b) Mapping the Data Attributes'.

(4) Manual Configuration of the Subscription

This is the most basic method. In the *GOOSE Configuration* module, add a new GOOSE (Fig. 9) and enter the parameters (GOOSE control block reference, GOOSE ID, Application ID, Broadcast MAC address) manually.

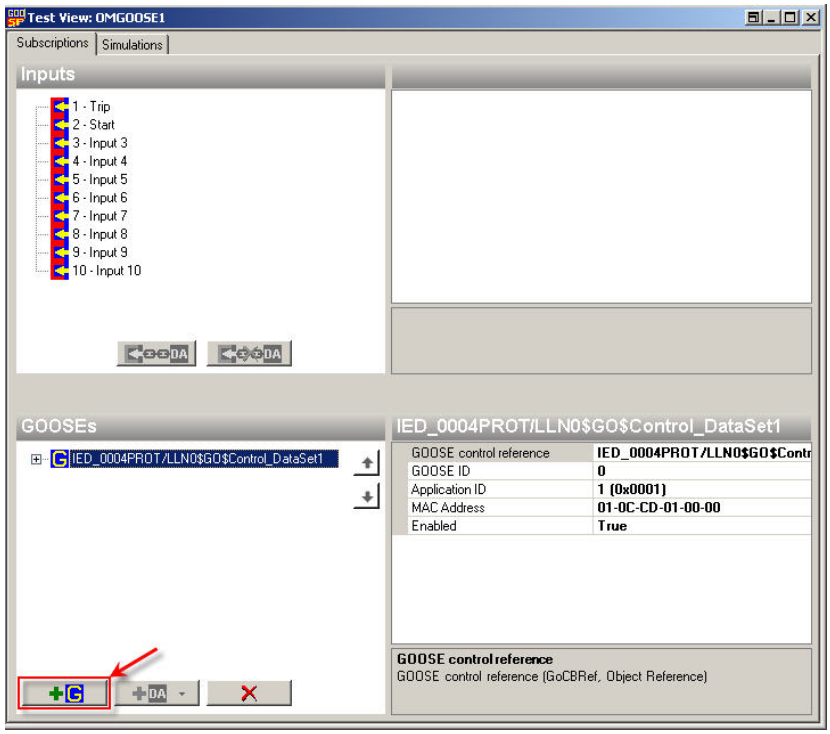

Fig. 9 Specifying GOOSE parameters for subscription

All items must match exactly; any typo will lead to an invalid subscription.

For the mapping, the dataset with the attributes is needed. To properly relate the data items with the binary inputs, the content of the GOOSE dataset needs to be specified as well. This is done by adding attributes to the dataset.

Expand the GOOSE, select 'Dataset' and click the "add attribute" button: 

By adding attributes, the dataset is populated. The structure of the dataset must be exactly resembled as sent out from the relay (see Fig. 10). Even items which are not intended to be mapped must be specified, since the test set needs these placeholders to find the proper position of the other values.

The attribute names are purely descriptive and do not need to match the objects references of the attributes correctly for a valid subscription.

To facilitate the setup of the bit strings for the quality data, the default length of the BitString type is 13.

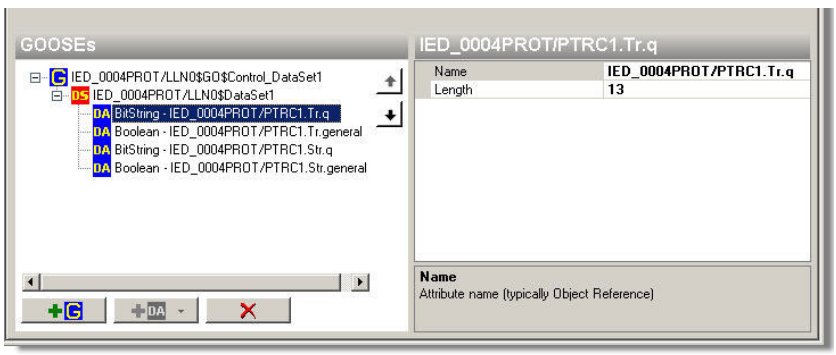

Fig. 10 Specifying the dataset

The type DoubleBit

The type DoubleBit is as special representation of a BitString[2].

The BitString[2] type is typically used for status indicators for switch positions etc.

While the bits of the BitString have to be mapped and treated individually, the DoubleBit provides a simpler handling of the data in most cases, because it can be mapped to a binary input/output as a whole.

Since the BitString[2] can assume more values (4) than a binary input/output can represent (2), a special mapping is performed.

Binary Inputs:

A DoubleBit value of [01] sets the binary input to inactive.

A DoubleBit value of [10] sets the binary input to active.

The DoubleBit values [00] and [11] do not change the status of the binary input.

Binary Outputs:

An inactive binary output sets the DoubleBit to [01].

An active binary output sets the DoubleBit to [10].

The DoubleBit values [00] and [11] are never used.

The GOOSE Configuration module uses DoubleBits by default when a BitString[2] is found in a datset.

If the bits need to be accessed individually (e.g. to make explicit mappings to 52A/52B), the Doublebit can be replaced by a BitString[2] (deleting the Doublebit and creating a BitString[2] in the same place).

Integer and Unsigned Data Attributes

Values contained in Integer and Unsigned DA (/Data Attributes) are compared against the value set in the "Value" property of the mapping.

Binary Inputs:

If the values match, the binary input becomes active.

If the values do not match, the binary input becomes inactive.

Binary Outputs:

When the binary output is active, the set "Value" is transmitted in the DA.

When the binary output is inactive, the "Complemented Value" is transmitted in the DA.

The "Complemented Value" is computed by bitwise complimenting the 32 bit value.

The configuration of the test set can then be continued with '(1b) Mapping the Data Attributes'.

Determining the GOOSE Details

For setting up the subscriptions, a number of GOOSE parameters and the exact structure of the dataset must be known. When using SCL files, all information is already provided.

When using GOOSE sniffing or manual setup, some details have to be looked up separately.

To find these details, several different options are outlined below.

The first option shown is capturing the data "online" from the GOOSE information actually sent out from the configured relay.

One way is to use the *OMICRON IEDScout*, which shows the actual state of a received GOOSE message. The relevant GOOSE parameters and the actual content of the dataset are monitored, as shown in the screenshot below:

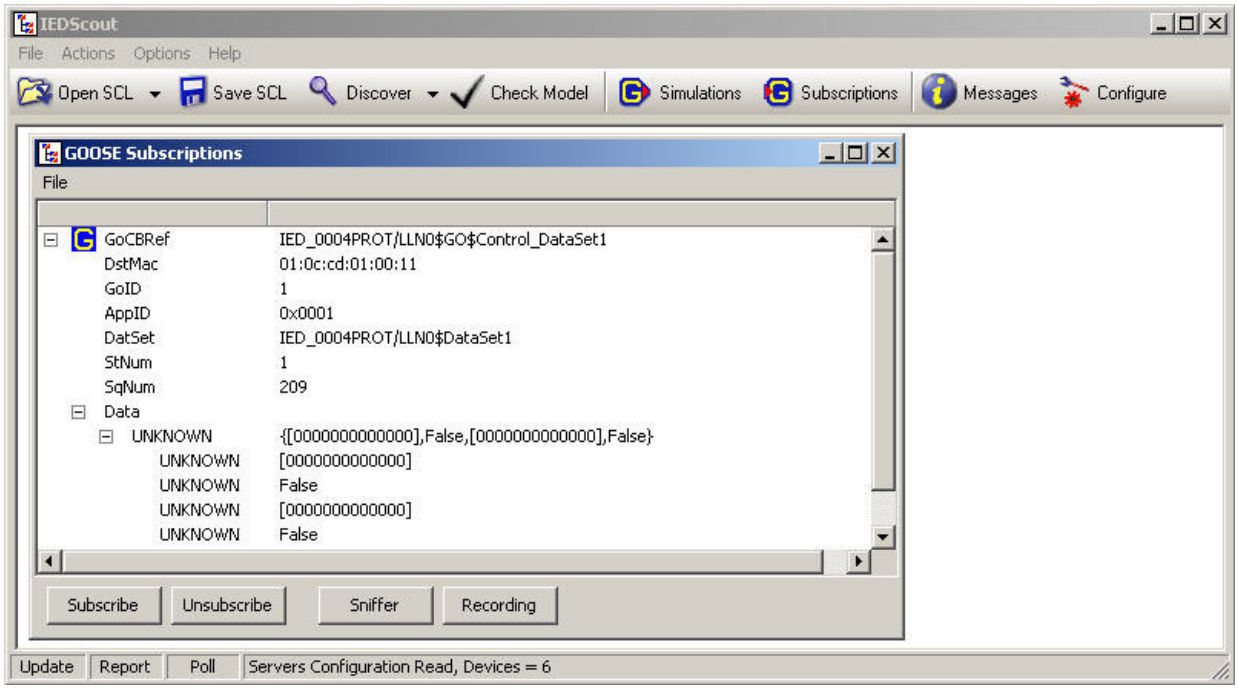

Fig. 11 Display of monitored GOOSE message

The "Data" field shows the contents of the dataset. Surprisingly, the quality information (the bit strings [0000000000000]) comes in front of the related status values (the booleans). It is also obvious that the times stamps are not included in the datasets.

With asserting fault values to the relay and observing the resulting changes of the status values, it can be determined that the first status value is the trip and the second is the start information.

Other parameters necessary for the subscription can be also obtained from the monitor display:

- the GOOSE control block reference (GoCBRef),
- the multicast MAC address (DstMac)
- the GOOSE ID (GoID),
- the GOOSE Application ID (AppID), and
- the dataset reference (DatSet).

With simple GOOSE sniffing, there is also no further information about the items in the dataset.

Another option is to obtain the GOOSE information from the relay's self description or the configuration file for the relay. With the *OMICRON IEDScout*, both methods work similar. In this case, the system configuration description (SCD file) is loaded, which is produced by IED system configuration software and select the "IED_0004".

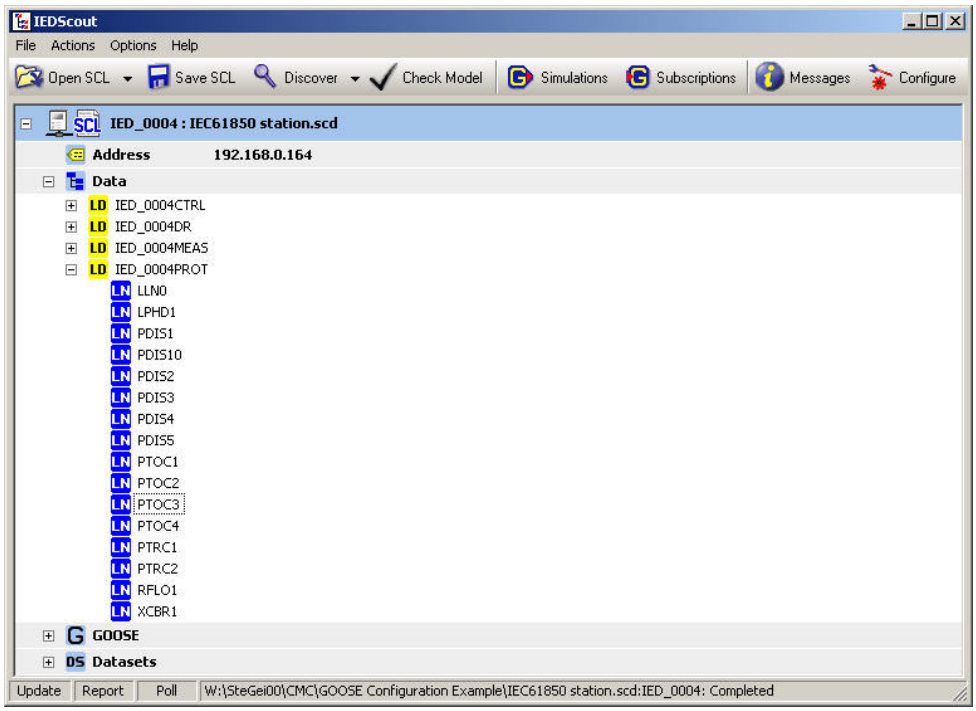

Fig. 12 Logical Devices and Logical Nodes of the Relay

When using the self description via the "Discover" function, the screen would look very similar, just the path to the filename would be replaced by the configured relay (server) name.

The concerned GOOSE control block that holds the required information is in the logical node zero (LLN0) of the logical device for the protection functions "PROT".

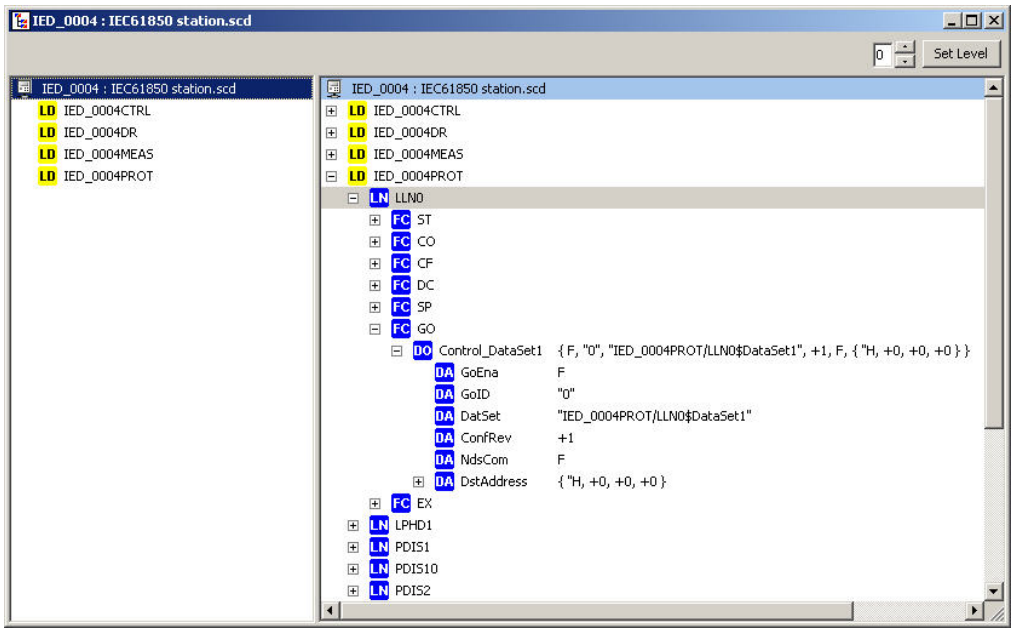

Fig. 13 GOOSE control block in LLN0 of logical device PROT

The object reference of the GOOSE control block, which is required for the configuration in the GOOSE module, can be easily derived by concatenating the object names. In this case, this delivers: "IED_0004PROT/LLN0GOControl_DataSet1".

Further details (and additional functions, which are not used here) can be found in dedicated dialogs for the GOOSE control blocks and datasets as shown below. Please note that the delimiter "$" is often seen in this context, while a "." would be expected from the IEC 61850 syntax. The "$" is the delimiter in the messages on the network, and the data are displayed just as they are delivered by the protocol.

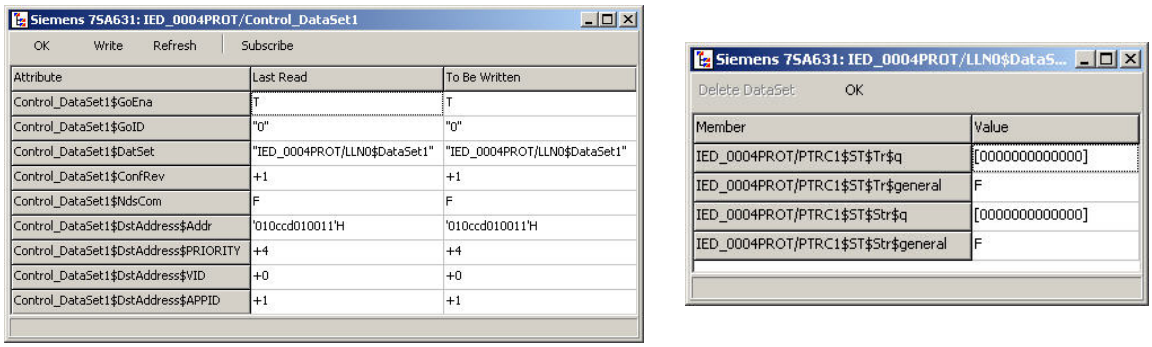

Fig. 14 Detail views of GOOSE control block and Dataset

The remarkable thing when using the browser is the dataset view on the right of Fig. 14. The full meaning of the items in the dataset is explicitly given, no guessing and trying with pick-up and trip to identify the items is necessary.

From the information delivered, using the self description or configuration data is the preferred method, even if it might be applicable less often than just capturing the actual traffic from the network. On the other hand, working from the configuration files allows completely offline preparation of the GOOSE configurations.

Performing the Tests

From the test module's view, the binary inputs are configured in the hardware configuration and applied for the testing as usual.

Using *QuickCMC*, the mapping of the binary signals can be easily verified. Feeding heavily unbalanced currents to the relay, a differential trip can be forced. Binary input 1 is active through the differential protection trip and binary input 2 is active since the contact input is short circuited at the relay. The binary input indicators show the received status.

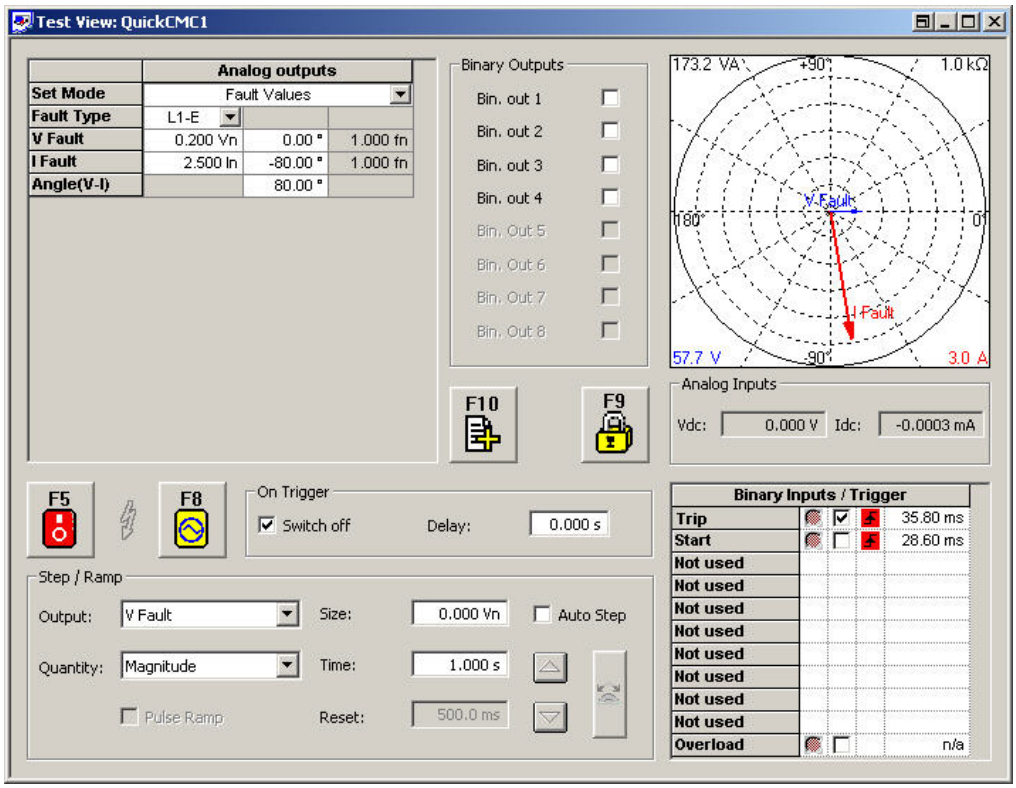

Fig. 15 *QuickCMC* User Interface

Of course, more sophisticated test modules like *Advanced Distance* and so on can be used as well.

Sending Binary Output Status via GOOSE

The test set can also simulate IEDs and send their GOOSEs on the network. The values in the GOOSE datasets are controlled by mapping binary outputs to them.

To illustrate how the IED simulation feature can be used, the 52A and 52B outputs of a CB simulation are sent out via GOOSE. For information how to set up a CB simulation in the Test Set, refer to the related documentation.

The two binary outputs are mapped to the individual bits in a two-bit bit string. Such data types (effectively holding enumerations) are frequently used in IEC 61850, e.g. in the logical node for a CB (XCBR) for the status value (stVal) of the breaker position (Pos).

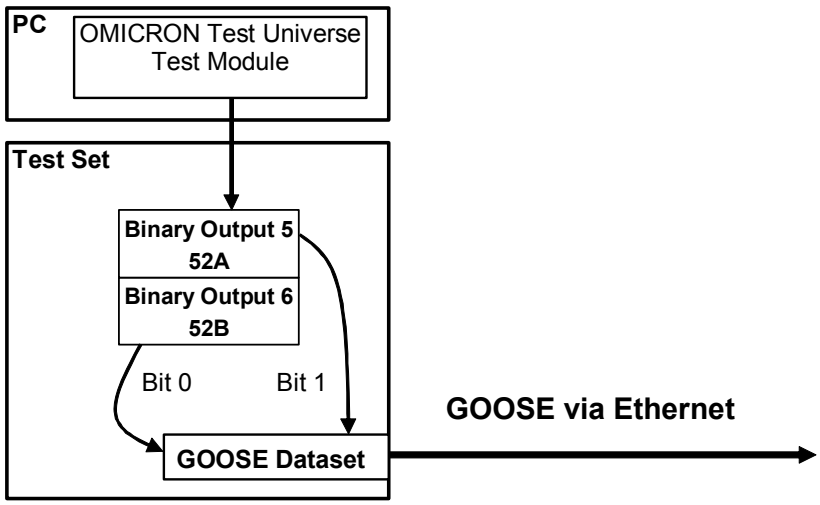

Fig. 16 Mapping of the CB aux contacts of the CB simulation to a two-bit status value in the GOOSE

The test set provides the eight binary outputs (four contact outputs and four transistor outputs) and up to 360 virtual outputs for the mapping. The mapped data can go into up to 128 simulated GOOSEs.

Configuration of the Simulation

Again, the GOOSE parameters and the GOOSE dataset have to be set up first. For sending GOOSE, some more parameters than for the subscription have to be specified. Some fields were just left with the defaults. If the existing SCL file contains a 'CB GOOSE', it can be taken for simulating (proceed as described in method (1) 'Using the SCL file').

In a real substation, the devices are only prepared to receive GOOSE from devices which are part of the substation configuration. In this case, the test set will simulate a device which is part of the substation. The other subscribers will not be reconfigured and will receive the "same" GOOSE, this time published by the test set.

Start the *GOOSE Configuration* module and switch to the 'Simulations' tab to configure the outputs. In this example, we simulate a device ('TestSet') with a logical device ('CB') to assemble a valid GOOSE control reference. The settings in Fig. 17 below suggest that there is a device 'TestSet' with a logical device 'CB', which contains a logical node XCBR1 of type XCBR.

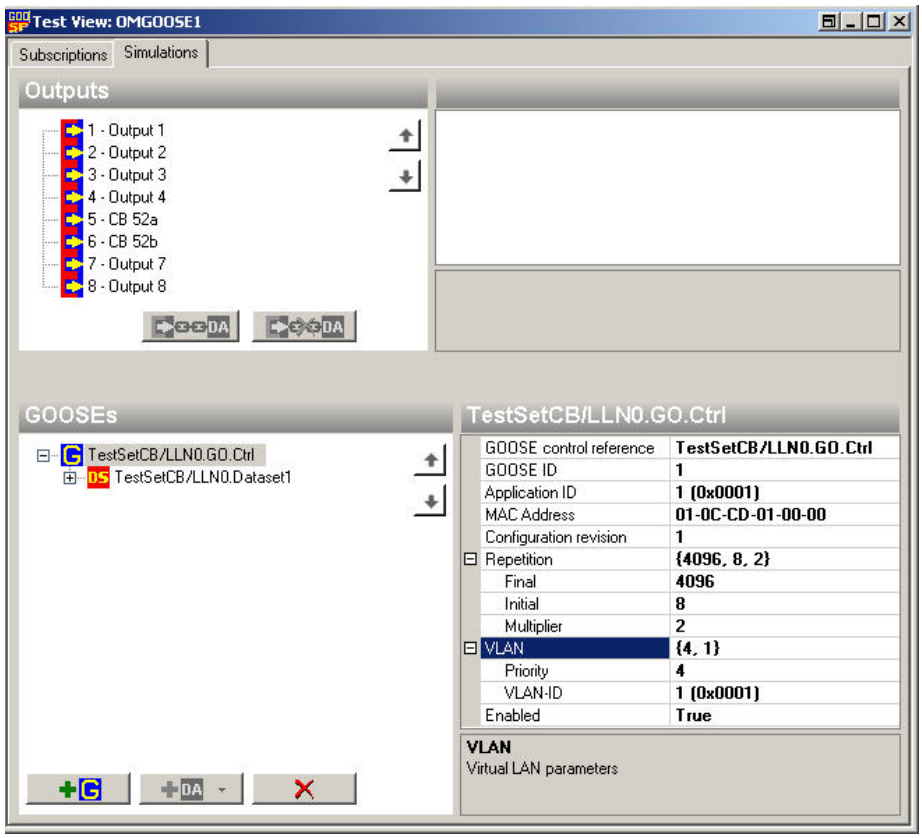

Fig. 17 Parameters for simulated GOOSE

The dataset contains simply one bit string with two bits, which shall represent the status values (52a and 52b auxiliary contacts) of the breaker.

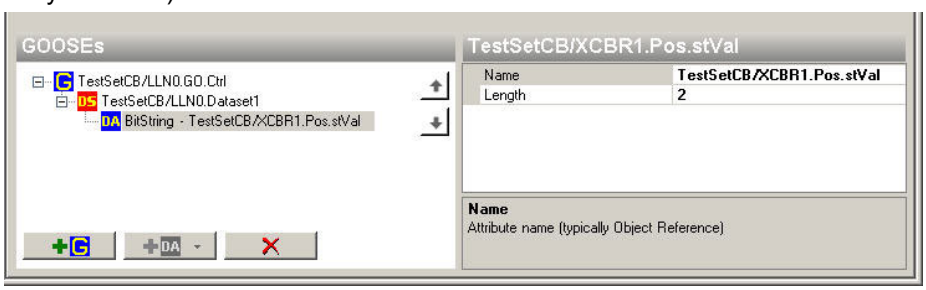

Fig. 18 Dataset with bitstring for 52a and 52b status

The CB simulation is configured to operate the first two transistor outputs (binary out 5 & 6) with the 52a and 52b signals.

After setting up the GOOSE and the dataset, these outputs can be mapped to the bits of the bit string.

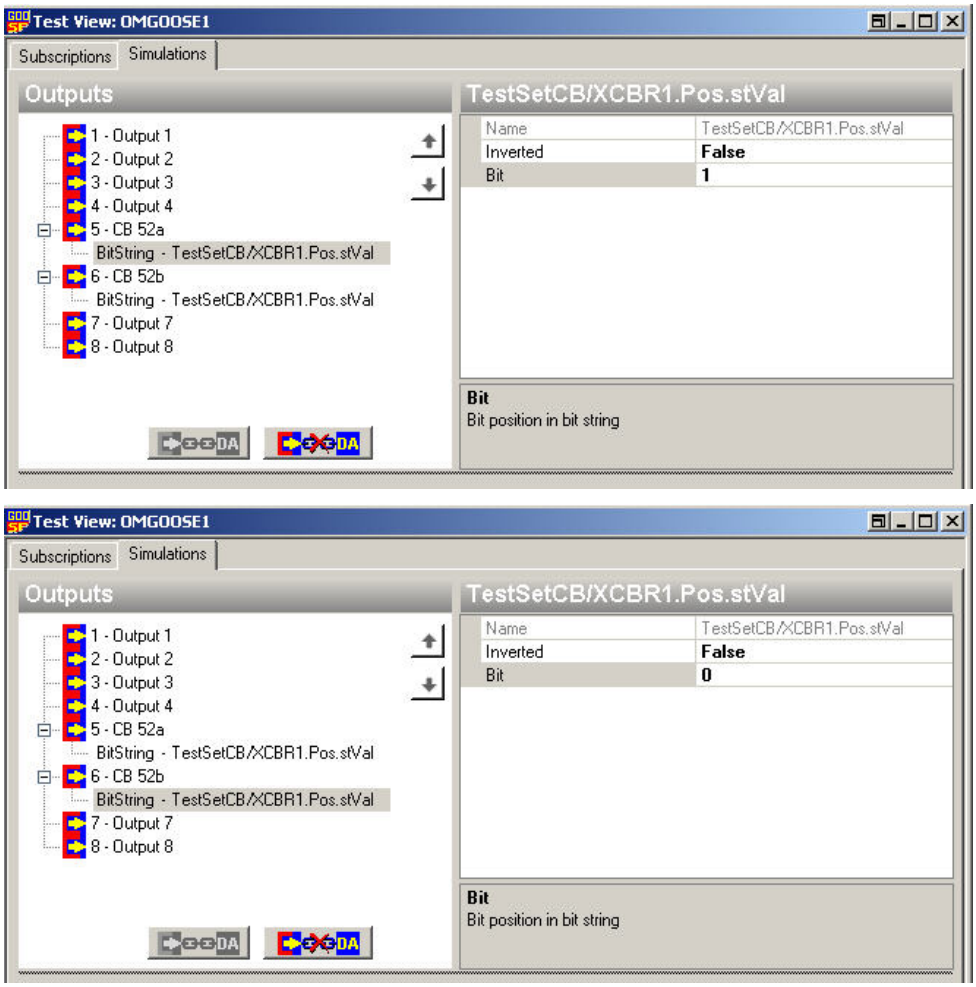

Fig. 19 Mapping the CB aux contacts into the GOOSE dataset

By pressing the start button, the GOOSE configuration is transferred to the test set. The progress bar indicates the progress of the configuration process. The "Test State" indicates if the new configuration was successfully applied.

By activating the binary outputs 5 and 6 from the CB simulation feature, the status of the CB auxilary contacts will be reflected in the published GOOSE from the Test Set. In the same way, other binary outputs can be used to operate other data items in the dataset of a simulated GOOSE.

Finally, the following figure shows how the mappings for subscriptions and simulations are reflected together in the report view of the GOOSE module.

GOOSE Subscriptions

Bin. Inp.	GOOSE Control Ref.	Attribute	Type	Value	Inverted
1 - Trip	IED_0004PROT/LLN0GOControl_DataSet1	IED_0004PROT/PTRC1.Tr.general	Boolean		no
2 - Start	IED_0004PROT/LLN0GOControl_DataSet1	IED_0004PROT/PTRC1.Str.general	Boolean		no

GOOSE Simulation

Bin. Out.	GOOSE Control Ref.	Attribute	Type	Value	Inverted
5 - CB 52a	TestSetCB/LLN0.GO.Ctrl	TestSetCB/XCBR1.Pos.stVal	BitString[2] - 1	0	no
6 - CB 52b	TestSetCB/LLN0.GO.Ctrl	TestSetCB/XCBR1.Pos.stVal	BitString[2] - 0	0	no

Fig. 20 Subscriptions and Simulations in the Report View

By changing the test module's report settings, even more detailed information about the GOOSE parameters and datasets can be displayed.