

IEC 61850 Application Note



History

Revision	Date	Description	Responsible
0.99	April 19 th 2016	First draft	DGA
1.00	May 10 th 2016	First release	MFA

Contents

1	Solution Overview	3
2	Anybus SG-gateway as IEC 61850 server.....	4
2.1	Communication section.....	4
2.2	IED section	4
2.2.1	Logical Device Power Quality	5
2.2.2	Logical Device Circuit Breaker.....	8
2.3	Datatype template section	10
3	Anybus SG-gateway as IEC 61850 client	11
3.1	Communication section.....	11
3.2	IED section for the client.....	11
3.3	IED section for the shadow server.....	12
3.4	Datatype template section	12
4	Use cases.....	13
4.1	Retrieve self-description	13
4.2	IEC 61850 server sending reports.....	14
4.3	IEC 61850 client receiving reports	15
4.4	IEC 61850 server receiving a command	18
4.5	IEC 61850 server publishing GOOSE.....	20
4.6	IEC 61850 client subscribing to GOOSE	21

1 Solution Overview

This application note describes the SCL files needed to run an IEC 61850 server and an IEC 61850 client on Anybus SG-gateways. The SCL files (a server file and a client file) are included in the .ZIP file together with this application note.

Six use cases are described in this application note:

1. Retrieve self-description.
2. IEC 61850 server sending reports.
3. IEC 61850 client receiving reports.
4. IEC 61850 server receiving a command.
5. IEC 61850 server publishing GOOSE.
6. IEC 61850 client subscribing to GOOSE.

2 Anybus SG-gateway as IEC 61850 server

In order to operate an Anybus SG-gateway as an IEC 61850 server, an SCL file describing the server's data model must be created and uploaded to the SG-gateway. The SCL file is an XML device description file holding the information that the gateway exposes to an IEC61850 client.

```
<?xml version="1.0"?>
<SCL xmlns="http://www.iec.ch/61850/2003/SCL"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.iec.ch/61850/2003/SCL SCL.xsd">
  <Header version="0" id="0"/>
  <Private>ICD file for demo on HMS Anybus SG-gateway WEB-PLC.</Private>
  <Private>This file is used for Example 1 retrieving self-description.</Private>
  <Private>This file is used for Example 2 cyclic reporting of measurements.</Private>
  <Private>This file is used for Example 4 controlling XCBR.Pos and event reporting of XCBR.Pos status change.</Private>
  <Private>This file is used for Example 5 GOOSE publishing on XCBR.Pos status change.</Private>
  <Private>All four examples are modelled into ONE Anybus SG Gateway.</Private>
  <Private>Documentation of examples see application note</Private>
  <Communication>
  <IED type="Anybus SG Gateway" name="PQSrv" manufacturer="HMS" desc="Demo of 61850 server SCL file">
  <DataTypeTemplates>
</SCL>
```

The SCL file is composed by three sections: [Communication](#), [IED](#) and [DataTypeTemplates](#).

2.1 Communication section

The communication section holds the network settings (IP address, subnet mask, MAC address, etc...) for the device. **Please enter your device settings in the corresponding fields!**

```
<Communication>
  <SubNetwork name="SubNetworkName">
    <ConnectedAP apName="SubstationRing1" iedName="PQSrv">
      <Address>
        <P type="OSI-AP-Title">1,1,9999,1</P>
        <P type="OSI-AE-Qualifier">12</P>
        <P type="OSI-PSEL">00000001</P>
        <P type="OSI-SSEL">0001</P>
        <P type="OSI-TSEL">0001</P>
        <P type="IP">0.0.0.0</P>
        <P type="IP-SUBNET">255.255.255.0</P>
        <P type="IP-GATEWAY">192.168.1.10</P><!-- Enter the IP address of the Anybus SG gateway acting as 61850 server here -->
        <P type="MAC-Address">00-30-56-A9-8D-8A</P><!-- Enter the MAC address of the Anybus SG gateway acting as 61850 server here -->
      </Address>
      <GSE ldInst="LDCB" cbName="CBStatus_GOOSE">
        <Address>
          <P type="MAC-Address">01-0C-CD-01-00-37</P>
          <P type="VLAN-PRIORITY">4</P>
          <P type="VLAN-ID">0</P>
          <P type="APPID">0</P>
        </Address>
      </GSE>
    </ConnectedAP>
  </SubNetwork>
</Communication>
```

The element [GSE](#) (Generic Substation Events) holds the information related to GOOSE messages, which provides a fast and reliable mechanism of transferring time-critical event data over the entire local area network using multicast or broadcast services. In this example we have configured a GOOSE message for the logical device circuit breaker ([LDCB](#)) to be sent from the Anybus SG gateway.

2.2 IED section

The IED section is divided in the first level in the [Services](#) element and in the [AccessPoint](#) element. The services subsection configures all services offered by the Anybus SG-gateway, e.g. a maximum of 2 GOOSE messages, 5 configuration report controls and 10 configuration datasets. The access point represents the interface to the network, and includes the IEC61850 server itself. In our example the server physical device (i.e. the Anybus SG-gateway) includes two logical devices:

1. Logical device power quality [LDPQ](#)
2. Logical device circuit breaker [LDCB](#)

```

<IED type="Anybus SG Gateway" name="PQSrv" manufacturer="HMS" desc="Demo of 61850 server SCL file">
  <Services>
    <DynAssociation/>
    <GetDirectory/>
    <GetDataObjectDefinition/>
    <DataObjectDirectory/>
    <GetDataSetValue/>
    <SetDataSetValue/>
    <DataSetDirectory/>
    <ConfDataSet max="10"/>
    <ReadWrite/>
    <ConfReportControl max="5"/>
    <GetCBValues/>
    <GOOSE max="2"/>
  </Services>
  <AccessPoint name="SubstationRing1">
    <Server timeout="30">
      <Authentication password="false"/>
      <LDevice inst="LDPQ">
        <LDevice inst="LDCB">
          </Server>
        </AccessPoint>
      </IED>

```

2.2.1 Logical Device Power Quality

The logical device power quality is composed of three logical nodes: Logical node zero **LN0**, logical node physical device **LPHD** and logical node measurements **MMXU**.

```

<LDevice inst="LDPQ">
  <LN0 desc="Logical node zero" inst="" lnType="LLN0_0" lnClass="LLN0">
    <LN inst="0" lnType="LPHD_0" lnClass="LPHD" prefix="">
      <LN desc="Measurement" inst="1" lnType="MMXU_0" lnClass="MMXU" prefix="">
    </LN>
  </LN0>
</LDevice>

```

2.2.1.1 Logical node MMXU

For a better understanding we will start explaining the logical node MMXU. For our example we will assume that a Modbus device providing data about **voltage, current, power and frequency** is connected to the Anybus SG-gateway. In this case we have to navigate through the IEC61850 specification to find the logical nodes that best match this information types. The MMXU logical node contains the data objects PhV (phase voltage), A (phase current), TotW (total active power) and Hz (frequency). Therefore the MMXU logical node will be used to model this information.

MMXU class				
Data object name	Common data class	Explanation	T	M/O/C
Data objects				
<i>Measured and metered values</i>				
TotW	MV	Total active power (total P)		O
TotVAr	MV	Total reactive power (total Q)		O
TotVA	MV	Total apparent power (total S)		O
TotPF	MV	Average power factor (total PF)		O
Hz	MV	Frequency		O
PPV	DEL	Phase to phase voltages (VL1, VL2, ...)		O
PNV	WYE	Phase to neutral voltage		O
PhV	WYE	Phase to ground voltages (VL1ER, ...)		O
A	WYE	Phase currents (IL1, IL2, IL3)		O

This structure is then replicated in the SCL file instantiating (`inst="1"`) the `MMXU_0` logical node type belonging to the class `MMXU`. The type `MMXU_0` is a derived version from class `MMXU` holding in this case just the optional data objects `PhV`, `A`, `TotW` and `Hz`.

```
<LN desc="Measurement" inst="1" lnType="MMXU_0" lnClass="MMXU" prefix="">
  <DOI name="TotW" desc="Measured total active power value">
    <SDI name="units">
      <DAI name="SIUnit">
        <Val>W</Val>
      </DAI>
    </SDI>
    <SDI name="mag">
      <DAI name="f">
        </DAI>
      </SDI>
    <DAI name="q">
      </DAI>
    <DAI name="t">
      </DAI>
    </DOI>
  <DOI name="Hz" desc="Measured frequency value">
  </DOI>
  <DOI name="PhV" desc="Phase to ground related measurement values of a three phase Star">
  </DOI>
  <DOI name="A" desc="Phase current related measurement values of a three phase Star">
  </DOI>
</LN>
```

Having a closer look to the data object `TotW` belonging to common data class `MV` (measured values), this element includes the data attributes `units`, magnitude `mag`, quality `q` and timestamp `t`. The last three attributes are mandatory in the `MV` class definition, while the `units` attribute is optional.

MV class					
Data attribute name	Type	FC	TrgOp	Value/Value range	M/O/C
DataName	Inherited from GenDataObject Class or from GenSubDataObject Class (see IEC 61850-7-2)				
DataAttribute					
measured attributes					
instMag	AnalogueValue	MX			O
mag	AnalogueValue	MX	dchg, dupd		M
range	ENUMERATED	MX	dchg	normal high low high-high low-low	O
q	Quality	MX	qchg		M
t	TimeStamp	MX			M
substitution and blocked					
subEna	BOOLEAN	SV			PICS_SUBST
subMag	AnalogueValue	SV			PICS_SUBST
subQ	Quality	SV			PICS_SUBST
subID	VISIBLE STRING64	SV			PICS_SUBST
blkEna	BOOLEAN	BL			O
configuration, description and extension					
units	Unit	CF	dchg	see Annex A	O
db	INT32U	CF	dchg	0 ... 100 000	O
zeroDb	INT32U	CF	dchg	0 ... 100 000	O
sVC	ScaledValueConfig	CF	dchg		AC_SCAV
rangeC	RangeConfig	CF	dchg		GC_CON_range
smpRate	INT32U	CF	dchg		O
d	VISIBLE STRING255	DC		Text	O
dU	UNICODE STRING255	DC			O
cdcNs	VISIBLE STRING255	EX			AC_DLND_A_M
cdcName	VISIBLE STRING255	EX			AC_DLND_A_M
dataNs	VISIBLE STRING255	EX			AC_DLND_M
Services					
As defined in Table 29.					

The magnitude **mag** data attribute has the datatype **AnalogueValue**, which is defined either as a 32 bit integer or a 32 bit floating point value. In our SCL file we are using the floating point version.

AnalogueValue type definition			
Attribute name	Attribute type	Value/Value range	M/O/C
<i>i</i>	INT32	integer value	GC_1
<i>f</i>	FLOAT32	floating point value	GC_1

The other data objects **PhV**, **A** and **Hz** have a similar internal structure.

2.2.1.2 Logical node zero

The logical node zero is a virtual node defining the communication mechanisms used by the logical device. IEC 61850 follows a client-server architecture, therefore a client can request data from the server at any time and poll the server periodically. In order to optimize bandwidth usage, more sophisticated communication mechanisms are available in IEC 61850.

```

<LN0 desc="Logical node zero" inst="" lnType="LLNO_0" lnClass="LLNO">
  <DataSet name="PQMeas">
    <FCDA lnClass="MMXU" fc="MX" doName="PhV.phsA.cVal.mag.f" lnInst="1" prefix="" ldInst="LDPQ"/>
    <FCDA lnClass="MMXU" fc="MX" doName="A.phsA.cVal.mag.f" lnInst="1" prefix="" ldInst="LDPQ"/>
    <FCDA lnClass="MMXU" fc="MX" doName="TotW.mag.f" lnInst="1" prefix="" ldInst="LDPQ"/>
    <FCDA lnClass="MMXU" fc="MX" doName="Hz.mag.f" lnInst="1" prefix="" ldInst="LDPQ"/>
  </DataSet>
  <ReportControl name="PQM_RCB" desc="Unbuf RCB" datSet="PQMeas" intgPd="5000" confRev="1" rptID="PQValues">
    <TrgOps period="true" dupd="false" qchg="false" dchg="false"/>
    <OptFields reasonCode="true" dataSet="true" timeStamp="true" seqNum="true"/>
    <RptEnabled max="2">
      <ClientLN iedName="MyClient" lnClass="IHMI" lnInst="1" prefix="" ldInst="none"/>
    </RptEnabled>
  </ReportControl>
  <DOI name="NamPlt">
    <DOI name="Mod">

```

Multiple data signals can be grouped in a **DataSet**, which will be sent out under certain circumstances defined in the **ReportControl** associated to it. In our sample SCL file, the dataset **PQMeas** includes the data objects voltage **PhV**, current **A**, power **TotW** and **Hz**. The associated report control **PQM_RCB** (report control is linked to the dataset through the attribute **datSet="PQMeas"**) defines the trigger option periodic for this dataset (**TrgOps period="true"**), i.e. it will be sent out every 5 seconds (**intgPd="5000"**).

2.2.1.3 Logical node physical device

The logical node physical device holds some information related to the hardware such as vendor name and serial number.

```

<LN inst="0" lnType="LPHD_0" lnClass="LPHD" prefix="">
  <DOI name="PhyNam">
    <DAI name="vendor" valKind="Set">
      <Val sGroup="0">HMS</Val>
    </DAI>
    <DAI name="serNum">
      <Val>003056A98D8A</Val>
    </DAI>
    <DAI name="model">
      <Val>Anybus SG with I/Os</Val>
    </DAI>
    <DAI name="location">
      <Val>Anywhere</Val>
    </DAI>
  </DOI>
  <DOI name="PhyHealth">
    <DOI name="Proxy">

```

2.2.2 Logical Device Circuit Breaker

The logical device circuit breaker **LDCB** is composed of three logical nodes: Logical node zero **LN0**, logical node physical device **LPHD** and logical node measurements **XCBR**.

```
<LDevice inst="LDCB">
  <LN0 desc="Logical node zero" inst="" lnType="LLNO_0" lnClass="LLNO">
  <LN inst="0" lnType="LPHD_0" lnClass="LPHD" prefix="">
  <LN desc="Circuit breaker" inst="1" lnClass="XCBR" lnType="XCBR_0" prefix="">
</LDevice>
```

2.2.2.1 Logical node XCBR

The logical node XCBR is the virtual representation of a circuit breaker. The main conceptual difference to the MMXU class is that it can be operated, i.e. the circuit can be opened or closed.

```
<LN desc="Circuit breaker" inst="1" lnClass="XCBR" lnType="XCBR_0" prefix="">
  <DOI desc="Enumerated status" name="Beh">
    <DAI name="stVal">
      <Val>on</Val>
    </DAI>
  </DOI>
  <DOI desc="Controllable double point" name="Pos">
    <DAI name="ctlModel">
      <Val>direct-with-normal-security</Val>
    </DAI>
    <SDI name="Oper">
      <DAI name="ctlVal">
      </DAI>
    </SDI>
    <DAI name="stVal">
    <DAI name="q">
    <DAI name="t">
  </DOI>
</LN>
```

The class XCBR has been defined in the specification as:

XCBR class				
Data object name	Common data class	Explanation	T	M/O/C
CBOpCap	ENS	Circuit breaker operating capability		O
POWCap	ENS	Point on wave switching capability		O
MaxOpCap	INS	Circuit breaker operating capability when fully charged		O
Dsc	SPS	Discrepancy		O
Measured and metered values				
SumSwARs	BCR	Sum of switched amperes, resettable		O
Controls				
LocSta	SPC	Switching authority at station level		O
Pos	DPC	Switch position		M
BlkOpn	SPC	Block opening		M
BlkCls	SPC	Block closing		M
ChaMotEna	SPC	Charger motor enabled		O
Settings				
CBTmms	ING	Closing time of breaker		O

In our example we are only using the position data object, which is defined as DPC (double point control) class with the mandatory data attributes status value (stVal), quality and timestamp. Since the position can be operated, the data attribute control model (ctlModel) is also present.

DPC class					
Data attribute name	Type	FC	TrgOp	Value/Value range	M/O/C
DataName	Inherited from GenDataObject Class or from GenSubDataObject Class (see IEC 61850-7-2)				
DataAttribute					
status and control mirror					
origin	Originator	ST			AC_CO_O
ctlNum	INT8U	ST		0..255	AC_CO_O
stVal	CODED ENUM	ST	dchg	intermediate-state off on bad-state	M
q	Quality	ST	qchg		M
t	TimeStamp	ST			M
stSeld	BOOLEAN	ST	dchg		O
opRcvd	BOOLEAN	OR	dchg		O
opOk	BOOLEAN	OR	dchg		O
tOpOk	TimeStamp	OR			O
substitution and blocked					
subEna	BOOLEAN	SV			PICS_SUBST
subVal	CODED ENUM	SV		intermediate-state off on bad-state	PICS_SUBST
subQ	Quality	SV			PICS_SUBST
subID	VISIBLE STRING64	SV			PICS_SUBST
blkEna	BOOLEAN	BL			O
configuration, description and extension					
pulseConfig	PulseConfig	CF	dchg		AC_CO_O
ctlModel	CtlModels	CF	dchg		M
sboTimeout	INT32U	CF	dchg		AC_CO_O
sboClass	SboClasses	CF	dchg		AC_CO_O
operTimeout	INT32U	CF	dchg		AC_CO_O
d	VISIBLE STRING255	DC		Text	O
dU	UNICODE STRING255	DC			O
cdcNs	VISIBLE STRING255	EX			AC_DLND_A_M
cdcName	VISIBLE STRING255	EX			AC_DLND_A_M
dataNs	VISIBLE STRING255	EX			AC_DLN_M
Services					
As defined in Table 39.					
parameters for control services					
Service parameter name		Service parameter type		Value/Value range	
ctlVal		BOOLEAN		off (FALSE) on (TRUE)	

2.2.2.2 Logical node zero

Event-driven communication can be accomplished in IEC 61850 with a dataset triggered on data change, or using GOOSE messages. In our SCL sample file we provide both examples for the activation of a circuit breaker.

The **DataSet CBStatus** includes the data objects position status **Pos.stVal**, quality **q** and timestamp **t**. The associated **ReportControl CB_RCB** (report control is linked to the dataset through the attribute **datSet="CBStatus"**) defines the trigger option data change for this dataset (**TrgOps dchg="true"**).

A **GSEControl** element must be defined in order to send out an event-driven GOOSE message. The dataset included in the GOOSE message is defined by the attribute **datSet="CBStatus"**. Additional configuration is done in the **GSE** element within the Communication section.

```

<LN0 desc="Logical node zero" inst="" lnType="LLN0_0" lnClass="LLN0">
  <DataSet name="CBStatus">
    <FCDA lnClass="XCBR" fc="ST" doName="Pos.stVal" lnInst="1" prefix="" ldInst="LDCB"/>
    <FCDA lnClass="XCBR" fc="ST" doName="Pos.q" lnInst="1" prefix="" ldInst="LDCB"/>
    <FCDA lnClass="XCBR" fc="ST" doName="Pos.t" lnInst="1" prefix="" ldInst="LDCB"/>
  </DataSet>
  <ReportControl name="CB_RCB" desc="Unbuf RCB" dataSet="CBStatus" intgPd="0" confRev="1" rptID="CBStatusValues">
    <TrgOps period="false" dupd="false" qchg="false" dchg="true"/>
    <OptFields reasonCode="true" dataSet="true" timeStamp="true" seqNum="true"/>
    <RptEnabled max="2">
      <ClientLN lcnName="MyClient" lnClass="IHMI" lnInst="1" prefix="" ldInst="none"/>
    </RptEnabled>
  </ReportControl>
  <GSEControl type="GOOSE" name="CBStatus_GOOSE" desc="GOOSE with circuit breaker status" dataSet="CBStatus" confRev="1" appID="CB_ID"/>
  <DOI name="NamFit">
    <DOI name="Mod">

```

2.2.2.3 Logical node physical device

The logical node physical device holds some information related to the hardware such as vendor name and serial number.

2.3 Datatype template section

The datatype template section includes all the type definitions used in the SCL file.

```

<DataTypeTemplates>
  <LNNodeType id="XCBR_0" lnClass="XCBR">
  <LNNodeType id="LLN0_0" lnClass="LLN0">
  <LNNodeType id="MMXU_0" lnClass="MMXU">
  <LNNodeType id="LPHD_0" lnClass="LPHD">
  <DOType id="phsA_0" desc="Complex measured value" cdc="CMV">
  <DOType cdc="DPC" desc="Controllable double point" id="DPC_0">
  <DOType id="SPC_0" cdc="SPC">
  <DOType id="SPS_0" cdc="SPS">
  <DOType id="INS_0" cdc="INS">
  <DOType id="INC_1" cdc="INC">
  <DOType id="INS_3" cdc="INS">
  <DOType id="SPC_2" desc="Controllable single point" cdc="SPC">
  <DAType id="SPCOperate_0">
  <DAType id="Originator_0">
  <DOType id="LPL_0" cdc="LPL">
  <DOType id="DPL_0" cdc="DPL">
  <DOType id="WYE_0" desc="Phase to ground related measurement values of a three phase Star" cdc="WYE">
  <DOType id="MV_0" desc="Measured value" cdc="MV">
  <DOType id="SPS_1" desc="Single point status" cdc="SPS">
  <DAType id="mag_0">
  <DAType id="origin_0">
  <DAType id="Oper_0">
  <DAType id="magSVC_0">
  <DAType id="units_0">
  <DAType id="cVal_0">
  <DAType id="sVC_0">
  <EnumType id="OrCat">
  <EnumType id="ctlModels">
  <EnumType id="Dbpos">
  <EnumType id="Mod">
  <EnumType id="SIUnit">
  <EnumType id="Health">
</DataTypeTemplates>

```

The logical nodes type definitions include all data objects used by each node, the data object definitions include all data attributes, and the data attributes type definitions are broken down until only basic well-known datatypes (e.g. boolean, integer or string) remain.

3 Anybus SG-gateway as IEC 61850 client

In order to operate an Anybus SG-gateway as an IEC 61850 client, an SCL file describing the client itself and all servers connected to it must be created and uploaded to the SG-gateway.

```
<?xml version="1.0"?>
<SCL xmlns="http://www.iec.ch/61850/2003/SCL"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.iec.ch/61850/2003/SCL SCL.xsd">
  <Header version="0" id="0"/>
  <Private>ICD file for demo on HMS Anybus SG-gateway WEB-PLC.</Private>
  <Private>This file is used for Example 3 receive Report of Measurements and GOOSE on XCBR.Pos status change.</Private>
  <Private>This model incorporates the server model as a shadow server.</Private>
  <Private>Documentation of examples see application note</Private>
  <Communication>
    <IED name="MyClient">
      <IED type="Anybus SG Gateway" name="PQSrv" manufacturer="HMS" desc="Demo of 61850 server SCL file">
        <DataTypeTemplates>
        </SCL>
```

The main difference to the server file is the fact that we have multiple **IED** elements, one for the client itself and one for each server connected to it.

3.1 Communication section

The communication section holds the network settings (IP address, subnet mask, MAC address, etc...) for the device. **Please enter your device settings in the corresponding fields!**

```
<Communication>
  <SubNetwork name="SubNetworkName">
    <ConnectedAP iedName="MyClient" apName="SubstationRing1">
      <Address>
        <P type="IP">192.168.1.40</P><!-- Enter the IP address of the Anybus SG gateway acting as 61850 client here -->
        <P type="IP-SUBNET">255.255.255.0</P>
        <P type="IP-GATEWAY">192.168.1.1</P>
        <P type="OSI-TSEL">00000001</P>
        <P type="OSI-PSEL">01</P>
        <P type="OSI-SSEL">01</P>
        <P type="MAC-Address">00-30-56-A9-8D-87</P><!-- Enter the MAC address of the Anybus SG gateway acting as 61850 client here -->
      </Address>
    </ConnectedAP>
    <ConnectedAP apName="SubstationRing1" iedName="PQSrv">
      <Address>
        <P type="OSI-AP-Title">1,1,9999,1</P>
        <P type="OSI-AE-Qualifier">12</P>
        <P type="OSI-PSEL">00000001</P>
        <P type="OSI-SSEL">0001</P>
        <P type="OSI-TSEL">0001</P>
        <P type="IP">192.168.1.10</P><!-- Enter the IP address of the 61850 server here -->
        <P type="IP-SUBNET">255.255.255.0</P>
        <P type="IP-GATEWAY">192.168.1.1</P>
        <P type="MAC-Address">00-30-56-A9-8D-8A</P><!-- Enter the MAC address 61850 server here -->
      </Address>
      <GSE ldInst="LDCB" cbName="CBStatus_GOOSE">
        <Address>
          <P type="MAC-Address">01-0C-CD-01-00-37</P>
          <P type="VLAN-PRIORITY">4</P>
          <P type="VLAN-ID">0</P>
          <P type="APPID">0</P>
        </Address>
      </GSE>
    </ConnectedAP>
  </SubNetwork>
</Communication>
```

In addition to the network setting of the client itself, there are additional **ConnectedAP** elements for each IED connected to the client as IEC 61850 servers. Please note that we have exact the same content inside the **ConnectedAP** element as in the server SCL file.

3.2 IED section for the client

The IED section for the client includes only an **AccessPoint** element with the logical node **IHMI**. This class represents a human-machine interface that can be used to model a client.

```
<IED name="MyClient">
  <AccessPoint name="SubstationRing1">
    <LN lnClass="IHMI" lnType="IHMI" inst="1" />
  </AccessPoint>
</IED>
```

3.3 IED section for the shadow server

This IED section is an exact copy of the IED section from the server file. It's used by the client to parse and understand the data coming from the server.

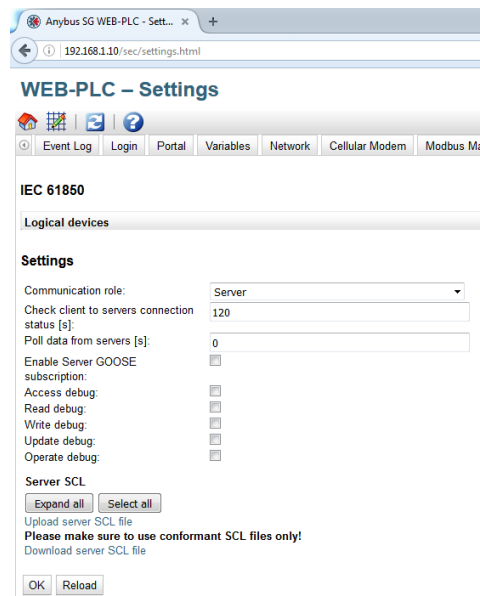
3.4 Datatype template section

The datatype template section includes all the type definitions used in the SCL file.

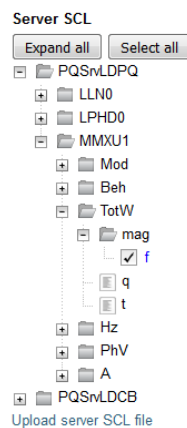
4 Use cases

4.1 Retrieve self-description

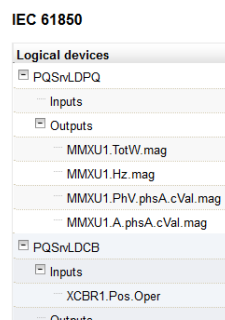
First we upload the server SCL file into the Anybus SG gateway. In order to do that, go to the IEC 61850 tab in the settings area and select the server communication role for the Anybus SG gateway. Then click on the “Upload server SCL file” link and upload the server SCL file.



Once the file is loaded, you will see the folder tree with the server information model. Select the signals that are going to be used in the application.

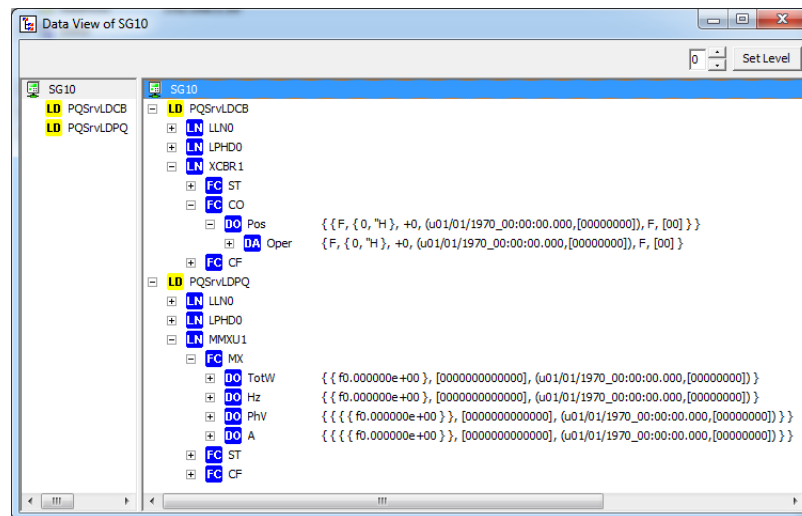


As you select the data objects for your application, they appear on the top under the logical devices section.



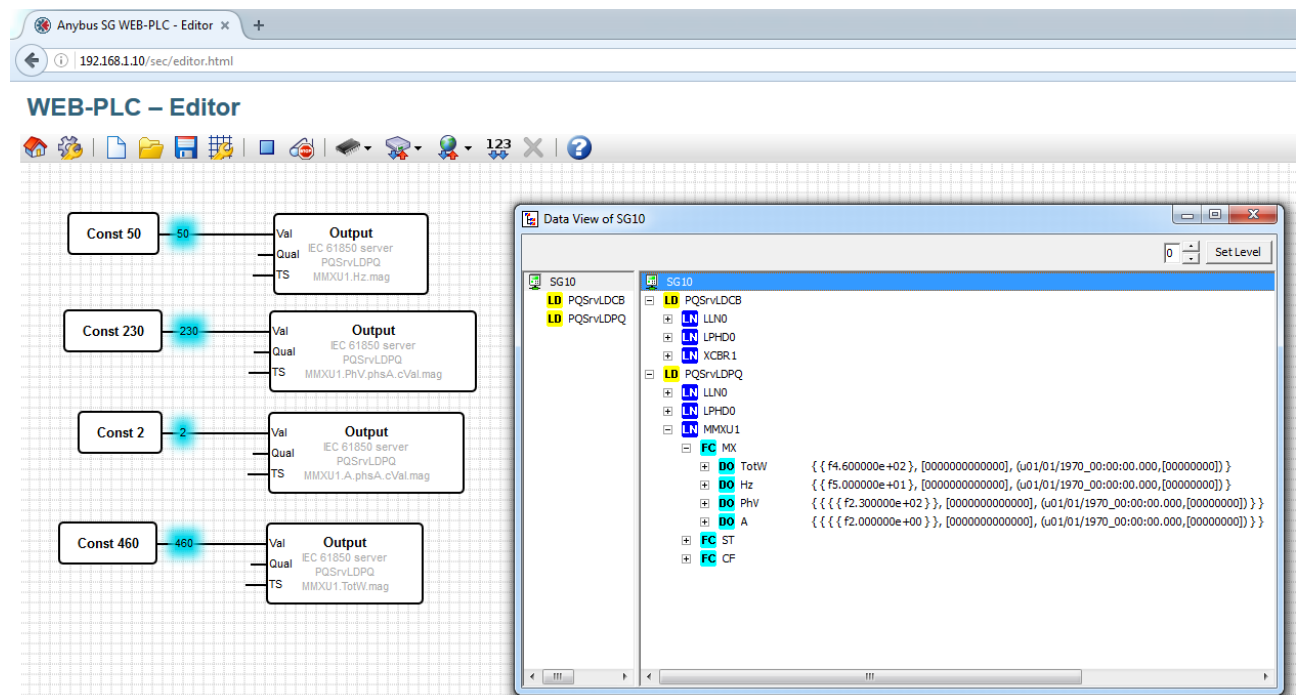
When you are finished, click the OK button to save your settings and reboot the device.

We will use the PC tool IEDScout as the IEC 61850 client. After entering the gateway's IP address, the client automatically retrieves the server data model.



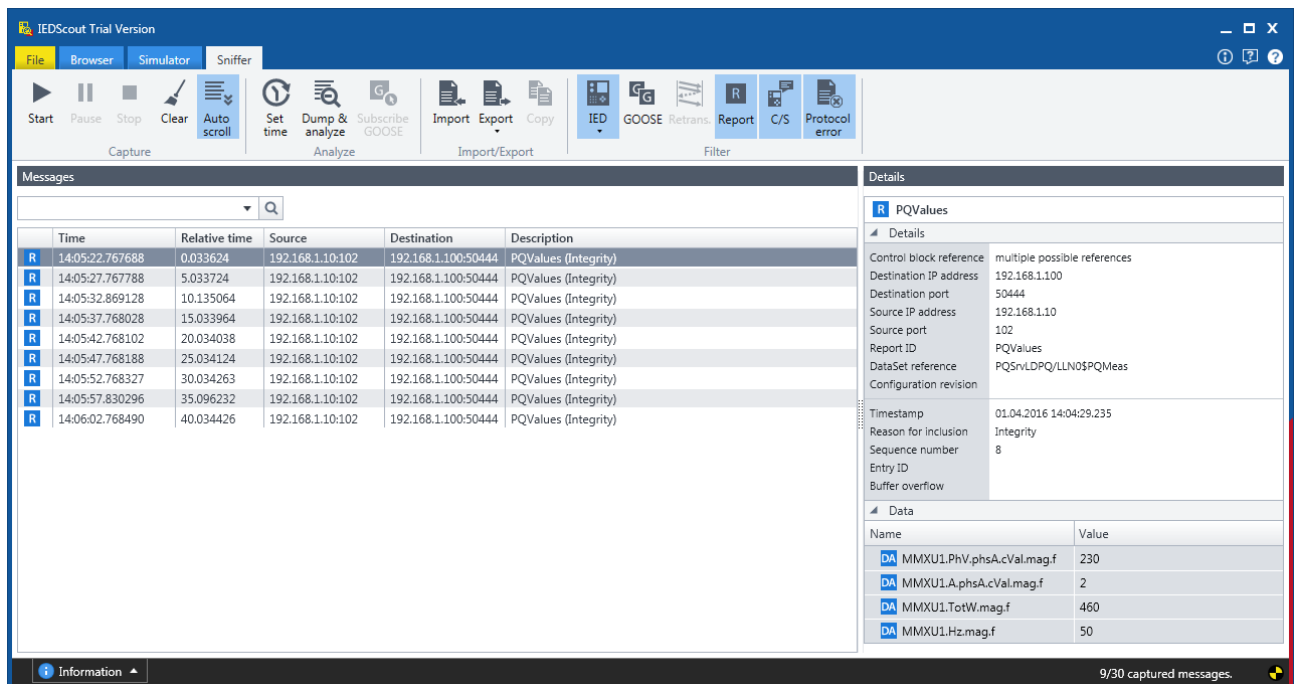
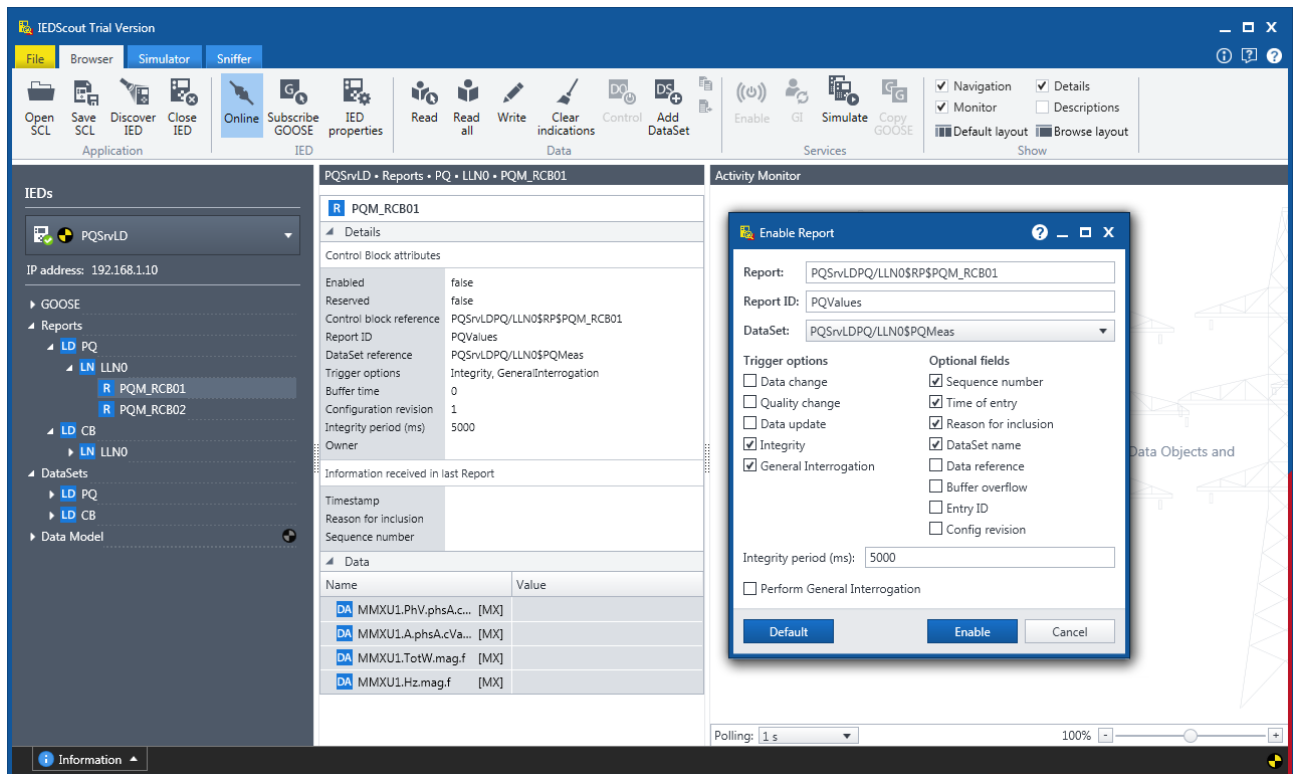
4.2 IEC 61850 server sending reports

We need to fill the information model with random values to be transmitted to the client. Therefore we will define some constants on the WEB-PLC editor and map them to the IEC 61850 signals.



The IEC 61850 client polls the server and receives all the current values. In addition to the polling option, the server's SCL file also defined periodic reports to be sent out every 5 seconds.

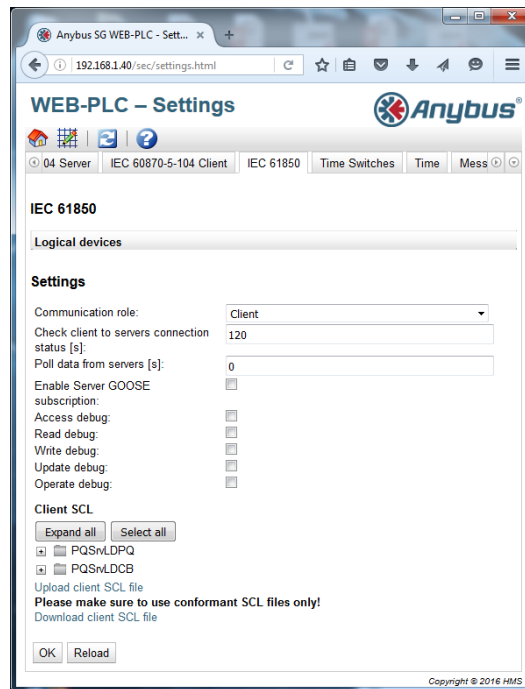
When we use a PC tool as IEC 61850 client, we need to enable the reports and then we can see them coming. If we are using another Anybus SG-gateway as IEC 61850 client, we can trace the traffic using Wireshark (we need port mirroring Switches).



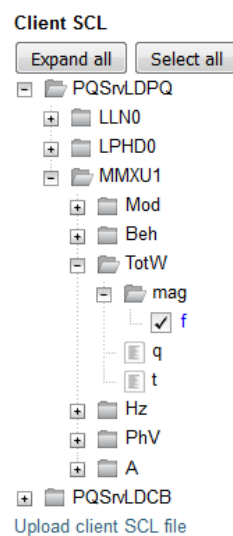
4.3 IEC 61850 client receiving reports

For this use case there is no additional setup to be done except for uploading the client SCL file into the Anybus SG-gateway acting as an IEC61850 client. In order to do that, go to the IEC 61850 tab in the settings area and select the client communication role for the Anybus SG gateway. Then click on the “Upload client SCL file” link and upload the client SCL file.

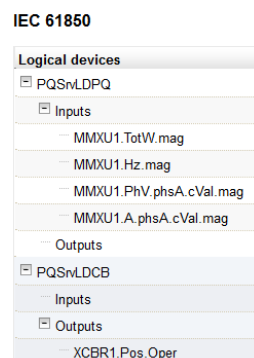
When the parameter “Poll data from servers [s]:” is set to zero, the client will not poll any server, i.e. all incoming data is either from reports sent out from the servers or from GOOSE messages.



Once the file is loaded, you will see the folder tree with the client information model. Select the signals that are going to be used in the application.

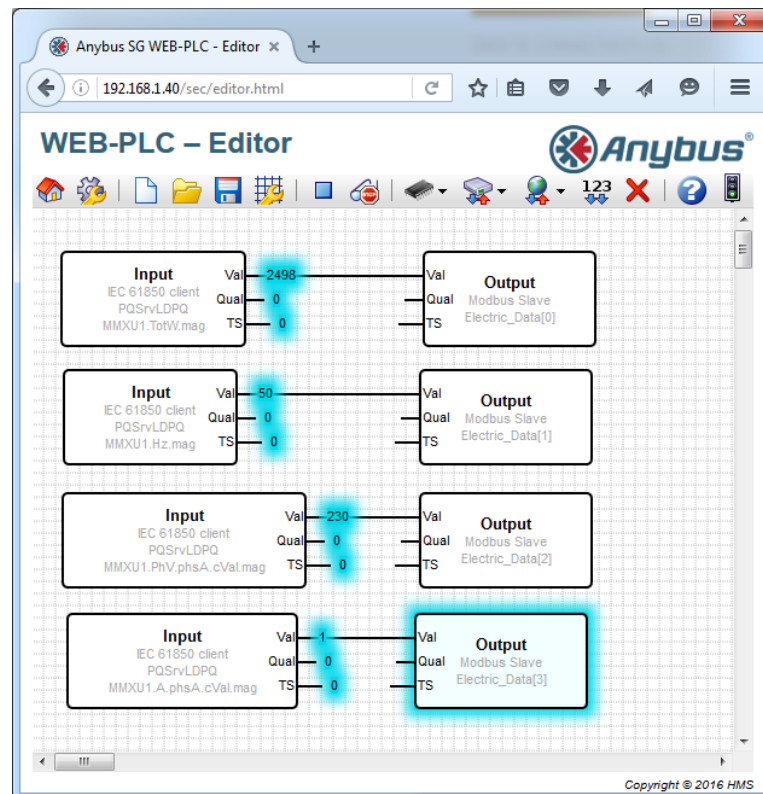


As you select the data objects for your application, they appear on the top under the logical devices section.



When you are finished, click the OK button to save your settings and reboot the device.

Now you can use these signals in your IEC 61850 client application. In this example we are mapping the four data objects from IEC 61850 into four Modbus TCP registers.



Now with a Modbus TCP client tool we can read out the registers and see the values.

Baseblock ComTest Pro for Modbus Devices

File Edit View Help

Serial Ethernet

Step 1

Protocol	IP Address	Port	Delay (ms)	Timeout (ms)
Modbus TCP	192.168.1.40	502	100	2000

Step 2

Device	Command	# Registers	Function
1	Read Holding Register(s)	4	3
Register	Write Single Holding Register		6
5	Write Holding Register(s)	1	16

☐ Loop Command
☒ Error Checking
☒ Show Error Dialog

Step 3

Start Stop Valid Response(s) Error Response(s) Timeout(s)

Read Registers Write Registers Raw Data Data Log

001..016:	2497d	50d	230d	1d	-	-	-	-	-	-	-	-	-	-	-	
017..032:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
033..048:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
049..064:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
065..080:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
081..096:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
097..112:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
113..125:	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

☐ Hexadecimal

Copy Data to Write Registers Copy Data to Log

www.baseblock.com Update Available

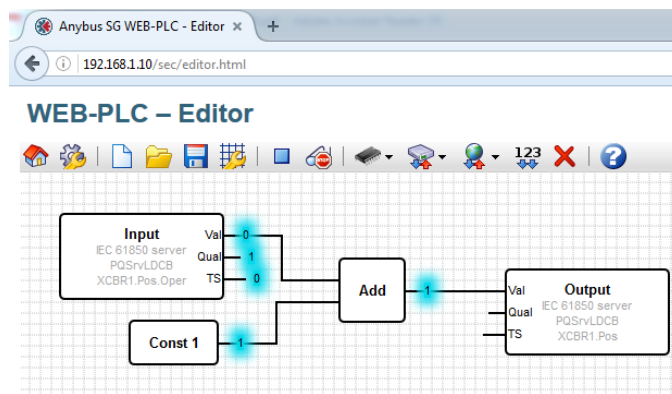
4.4 IEC 61850 server receiving a command

For this example we will use a circuit breaker, which is modelled in IEC 61850 with four possible states:

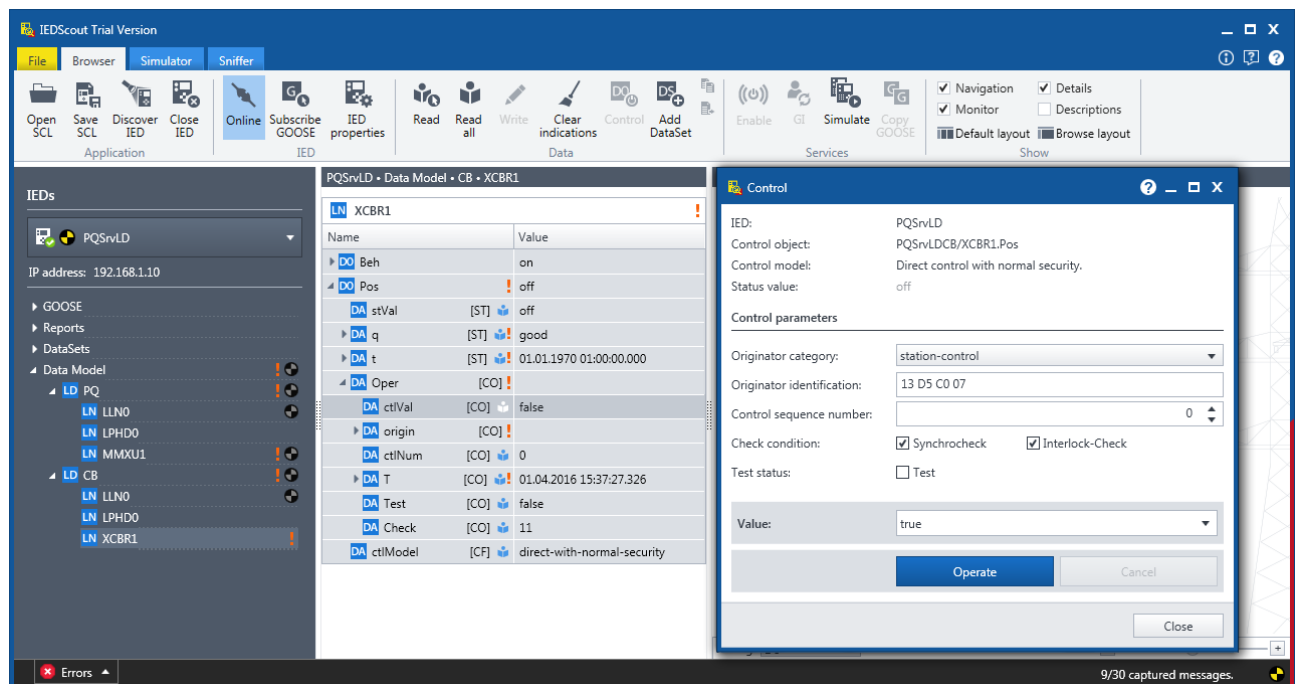
- 01: Off
- 10: On
- 00: Intermediate position (not a valid state)
- 11: Invalid

Each bit represent the measurement of a digital sensor in the open and closed position. When both sensors return a 0 (non presence) the switch is in an intermediate state. If both sensors return a 1 there is an error (switch cannot be in both positions simultaneously).

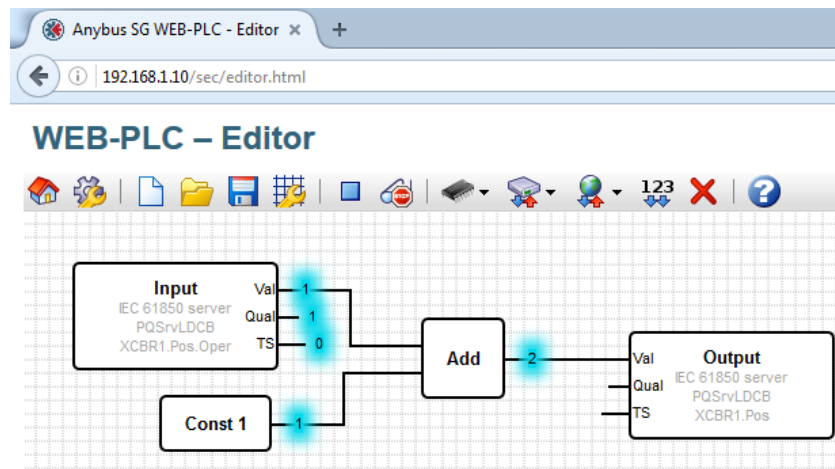
The XCBR.Pos.stVal signal represents the current position of the switch (on/off represented as decimal value with 2 or 1 respectively), while the XCBR.Pos.Oper.ctlVal indicates the set value as a boolean variable. Since there is an offset of one between the set value (true/false represented as 1 or 0 respectively) and the status value (on/off represented as 2 or 1 respectively), we build the application on the WEB-PLC adding 1 to the set value.



In this moment the circuit breaker is at the off position. From the IEC 61850 client in the PC we send the order to close the circuit breaker by changing the set value to true.



We see on the WEB-PLC application that the value of the position has changed to 2, i.e. the circuit breaker is on.



At the IEC61850 client we can also see the change from the circuit breaker.

IEDScout Trial Version

File | Browser | Simulator | Sniffer

Open SCL | Save SCL | Discover IED | Close IED | Online | Subscribe GOOSE | IED properties | Read | Read all | Write | Clear indications | Control | Add DataSet

Application | IED | Data

IEDs

PQSrvLD

IP address: 192.168.1.10

GOOSE

Reports

DataSets

Data Model

LD PQ

LN LLN0

LN LPHD0

LN MMXU1

LD CB

LN LLN0

LN LPHD0

LN XCBR1

PQSrvLD - Data Model - CB - XCBR1

LN	Name	Value
DO Beh	Beh	on
DO Pos	Pos	on
DA stVal	[ST]	on
DA q	[ST]	good
DA t	[ST]	01.01.1970 01:00:00.000
DA Oper	[CO]	
DA ctiVal	[CO]	true
DA origin	[CO]	
DA ctiNum	[CO]	0
DA T	[CO]	01.04.2016 15:48:44.298
DA Test	[CO]	false
DA Check	[CO]	11
DA ctiModel	[CF]	direct-with-normal-security

4.5 IEC 61850 server publishing GOOSE

We can generate GOOSE traffic creating a circuit breaker event. The server SCL file has defined a GOOSE message (see chapter 3.2.2.2) to be sent out on every change from the circuit breaker. Using an IEC 61850 tool such as IEDScout, we send the order to close the circuit breaker as explained in the previous section.

The screenshot shows the IEDScout Trial Version interface. The 'Messages' pane displays a list of captured GOOSE messages. The 'Details' pane shows the configuration for the selected message.

Time	Relative time	Source	Destination	Description
15:54:50.481608	0.000000	00:30:56:A9:8D:87	01:0C:CD:01:00:37	PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE
15:55:14.694203	24.212595	00:30:56:A9:8D:87	01:0C:CD:01:00:37	PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE
15:55:14.696092	24.214484	00:30:56:A9:8D:87	01:0C:CD:01:00:37	PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE
15:55:14.699254	24.217646	00:30:56:A9:8D:87	01:0C:CD:01:00:37	PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE
15:55:42.470987	51.989379	00:30:56:A9:8D:87	01:0C:CD:01:00:37	PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE
15:55:42.472874	51.991266	00:30:56:A9:8D:87	01:0C:CD:01:00:37	PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE
15:55:42.474789	51.993181	00:30:56:A9:8D:87	01:0C:CD:01:00:37	PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE

Details

Details

Control block reference: PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE

Destination MAC address: 01:0C:CD:01:00:37

Source MAC address: 00:30:56:A9:8D:87

Application ID: 0

GOOSE ID: CB_ID

DataSet reference: PQSVrLDCB/LLN0\$CBStatus

VLAN ID:

VLAN priority:

Needs commissioning: false

Configuration revision: 1

Simulation/Test: false

Entry time: 01.04.2016 15:54:21.082

Status number: 14

Sequence number: 0

Time allowed to live: 200

Number of DataSet entries: 3

Data

Name	Value
DA XCBR1.Pos.stVal	off
DA XCBR1.Pos.q	good
DA XCBR1.Pos.t	01.01.1970 01:00:00.000

After the first change of the circuit breaker (approx. 24 seconds after start), three GOOSE messages are sent out by the server holding the new position of the circuit breaker (Pos.stVal off), the quality (good) and timestamp.

The screenshot shows the IEDScout Trial Version interface. The 'Messages' pane displays a list of captured GOOSE messages. The 'Details' pane shows the configuration for the selected message.

Time	Relative time	Source	Destination	Description
15:54:50.481608	0.000000	00:30:56:A9:8D:87	01:0C:CD:01:00:37	PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE
15:55:14.694203	24.212595	00:30:56:A9:8D:87	01:0C:CD:01:00:37	PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE
15:55:14.696092	24.214484	00:30:56:A9:8D:87	01:0C:CD:01:00:37	PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE
15:55:14.699254	24.217646	00:30:56:A9:8D:87	01:0C:CD:01:00:37	PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE
15:55:42.470987	51.989379	00:30:56:A9:8D:87	01:0C:CD:01:00:37	PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE
15:55:42.472874	51.991266	00:30:56:A9:8D:87	01:0C:CD:01:00:37	PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE
15:55:42.474789	51.993181	00:30:56:A9:8D:87	01:0C:CD:01:00:37	PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE

Details

Details

Control block reference: PQSVrLDCB/LLN0\$GO\$CBStatus_GOOSE

Destination MAC address: 01:0C:CD:01:00:37

Source MAC address: 00:30:56:A9:8D:87

Application ID: 0

GOOSE ID: CB_ID

DataSet reference: PQSVrLDCB/LLN0\$CBStatus

VLAN ID:

VLAN priority:

Needs commissioning: false

Configuration revision: 1

Simulation/Test: false

Entry time: 01.04.2016 15:54:48.857

Status number: 17

Sequence number: 0

Time allowed to live: 200

Number of DataSet entries: 3

Data

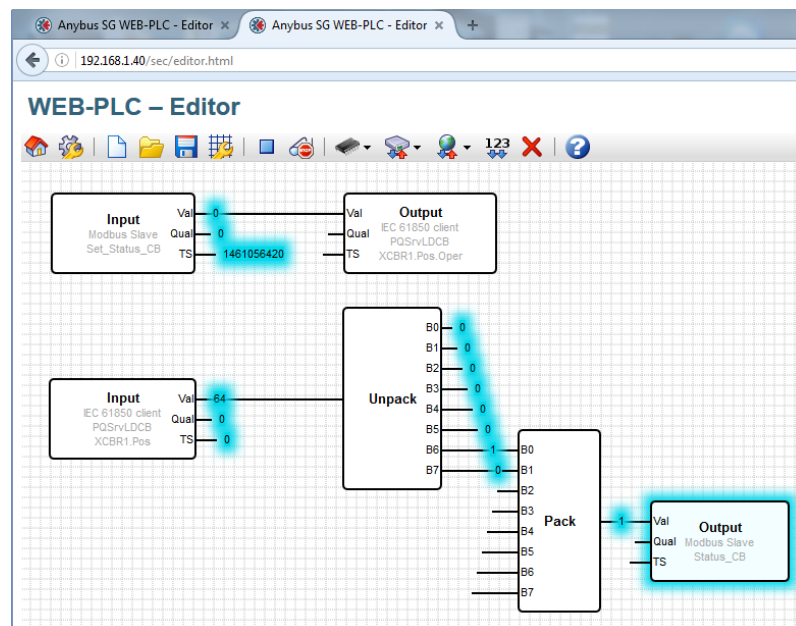
Name	Value
DA XCBR1.Pos.stVal	on
DA XCBR1.Pos.q	good
DA XCBR1.Pos.t	01.01.1970 01:00:00.000

On the next circuit breaker event (approx. 51 seconds after start), three GOOSE messages are sent out again by the server holding the new position of the circuit breaker (Pos.stVal on), the quality (good) and timestamp.

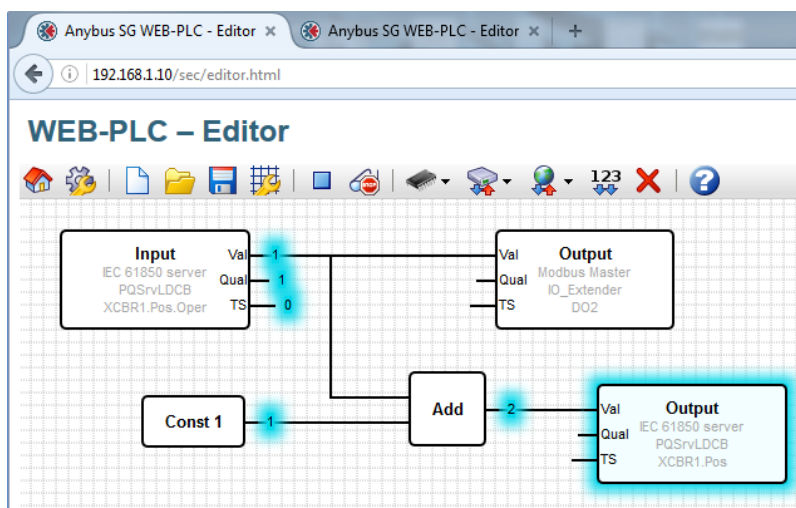
4.6 IEC 61850 client subscribing to GOOSE

For this use case there is no additional setup to be done except for uploading the client sample SCL file, which includes GOOSE mechanism, into the Anybus SG-gateway acting as an IEC61850 client. When the parameter “Poll data from servers [s]:” is set to zero, the client will not poll any server, i.e. all incoming data is either from reports sent out from the servers or from GOOSE messages.

In our example application we will configure the Anybus SG-gateway acting as an IEC 61850 client to be also Modbus TCP server. Using a PC-based Modbus TCP client tool, we will write the register “Set_Status_CB” and map it to the data object XCBR1.Pos.Oper to operate the circuit breaker on the server side.



When we activate the circuit breaker via Modbus on the client, the command is transmitted to the server and the operation is performed. The status of the circuit breaker XCBR1.Pos.StVal is then broadcasted via GOOSE messages because its value has changed.



Finally the GOOSE message is received by the IEC 61850 client and the circuit breaker's position XCBR1.Pos is updated from 1 (status off) to 2 (status on). Please read chapter 5.4 to learn more about the status values for circuit breakers. Since the bytes are swapped, we need to use the function blocks unpack and pack in order to assign the right value to the Modbus register holding the position status.

