

IpConv
Protocol Stack

Elcom90Initiator

ELCOM–90 Protocol, Initiator UE

IPCOMM GmbH
Gundstrasse 15
D-91056 Erlangen
Germany

Voice: +49 9131 92076 0
Fax: +49 9131 92076 10
Email: info@ipcomm.de

© IPCOMM GmbH 1994–2013
all rights reserved

Table of Contents

<u>Elcom90Initiator – General</u>	1
<u>Configuration</u>	1
<u>Import of information configuration from table</u>	1
<u>Startup</u>	1
<u>Implementation details</u>	1
<u>Elcom90Initiator – Normalized Address (NA)</u>	2
<u>Protocol specific information types</u>	3
<u>Internal module specific information</u>	3
<u>Elcom90Initiator – Logging</u>	4
<u>Logging level 1</u>	4
<u>Logging level 2</u>	5
<u>Logging level 4</u>	6
<u>Example of a logfile</u>	8
<u>Elcom90Initiator – Parameter</u>	10

Elcom90Initiator – General

The module described here implements the presentation layer, application layer and the Initiator part of the user element (UE) of the ELCOM–90 protocol.

Configuration

To configure the module the following steps have to be performed:

1. Adding at least one communication partner to the **partner** list.
2. Configuration of the communication parameters like the server IP address (this is also used as Acceptor address).
3. Configuration of the Initiator address.
4. Adding groups to the **group** list.
5. Adding information objects to the **object** list. Each information object can be assigned to several groups.

If the configuration is imported from table, step 5 can be omitted. This part will be derived from table.

Import of information configuration from table

Information configuration can be imported from table (e.g. MS Excel). For detailed description of the table file format see "ipConv Functional Specification".

The imported data is stored in the **import** parameter branch, which is automatically generated. The configuration of information objects can be performed manually and imported from table at the same time. In this case following points should be taken into consideration:

- Never mix both configuration methods for the same information object.
- As a rule only those information objects are defined manually which require exceptional treatment e.g. administrative indications. By means of import from table big amount of similar information objects can be imported more effectively.
- On repeated import from table configuration information imported before is removed and replaced by the new one.
- the imported configuration information can be modified manually. These changes will be lost on repeated imports.

To define information objects completely by means of import from a table following additional protocol–stack–specific column is required:

Key	Range	Description
/SRC.1.PSGROUP\$ /DST.1.PSGROUP\$	1..32767	The entry in this column assigns an information object to a group. The group number must be defined in the group list.

Startup

On startup of the module the following actions are executed:

1. The Initiator establishes all required associations, i.e. connection AA to AG.
2. The Initiator creates the groups in the Responder.
3. The Initiator adds information objects to the groups.
4. The Initiator gives permission for unsolicited data transfer to the Responder.
5. The Initiator sends a general scan to the Responder, the Responder sends the initial values back to the Initiator.
6. From now on, the Responder sends data spontaneously or cyclically to the Initiator. It depends on the used association, which functional unit (FU) is used for the data transfer.

Implementation details

For further information about the implementation details, please refer to the ELCOM implementation document (PID).

Elcom90Initiator – Normalized Address (NA)

Class	Direction	NA	Protocol Specific Information Type	Normalized Information Type	Description
Process Information	From Module to Node (from)	inf- <u>{PID}</u> - <u>{OBJID}</u>	StatusGroup	SI	Single point information
			StatusGroup	DI	Double point information
			MeasureGroup	MFV	Measured short float pointing value
			DiscreteGroup	MSV	Measured scaled value
			DiscreteGroup	MNV	Measured normalized value
	From Node to Module (to)	inf- <u>{PID}</u> - <u>{OBJID}</u>	BinaryCommandGroup	SC	Single command
			BinaryCommandGroup	DC	Double command
			AnalogueSetpointGroup	SPF	Setpoint command, short floating point
DigitalSetpointGroup			SP	Setpoint command	
Internal Information, connection specific	From Module to Node (from)	con- <u>{PID}</u> -connect		SI	Connection state to a specific partner
Internal Information, module specific	From Module to Node (from)	int-startup		SI	End of module start-up indication
	From Node to Module (to)	int-active		SI	Activate / Deactivate communication

Abbreviations

{PID}	Partner ID (1..255)
{OBJID}	Object Identifier ([-A-Za-z0-9_]{1,255})

Protocol specific information types

Group type	Description
Measure-group	Floating-point values
Status-group	3-state values (on/off/between)
Discrete-group	Two octet integer
Binary-command-group	2-state commands (on/off)
Analog-setpoint-group	Floating point values
Digital-setpoint-group	Two octet integer command values

Internal module specific information

int-startup

This single indication (SI) signals the module startup end with the value "APP". This happens after receiving general scan telegrams from all configured partners or after the timeout for the forced startup has elapsed. Normally this message is used to start up communication of high-order protocol modules.

int-active

This information is required for redundant operation of two or more protocol converters. If the configuration parameter "active" has the value "FALSE", the module starts up in passive state, i.e. no TCP connection is established. Receipt of this single indication with the value "APP" activates the module. The module then establishes the TCP connections to the partners. When this single indication is received with the value "DIS" and the module was previously active, it reverts to the passive state, i.e. all TCP connections are closed.

Elcom90Initiator – Logging

This protocol module distinguishes between three logging levels. The logging levels can be activated at the same time with the OR function.

Logging level	Description
0	no logging
1	hexadecimal representation of transport layer telegrams
2	decoded representation of presentation layer frames
4	decoded representation of application layer frames
6	levels 2 and 4
7	levels 1, 2 and 4

Logging level 1

The logging level 1 displays all the telegrams sent and received in hexadecimal presentation. The following syntax is used:

(1): {D} [{PID}/{FU}] <{DATA}>

Code	Value	Description
{D}		Transmission direction
	<<	Send direction
	>>	Receive direction
{PID}	1..255	Partner ID
{FU}	1..7	Functional Unit
{DATA}	00..FF	Telegrams sent and received in hexadecimal presentation

Logging level 2

The logging level 2 displays the PPDUs (Presentation Protocol Data Units) human-readable presentation. The following syntax is used:

```
(2): {D} [{PID}/{FU}] LEN={LEN} PDU-ID={PPDU} I-ADDR=<{ADDR}> A-ADDR=<{ADDR}> DATA=<{DATA}>
(2): {D} [{PID}/{FU}] LEN={LEN} PDU-ID={PPDU} I-ADDR=<{ADDR}> A-ADDR=<{ADDR}> RESULT={RESULT}
DATA=<{DATA}>
(2): {D} [{PID}/{FU}] LEN={LEN} PDU-ID={PPDU} REASON={REASON}
(2): {D} [{PID}/{FU}] LEN={LEN} PDU-ID={PPDU} RESULT={RESULT}
(2): {D} [{PID}/{FU}] LEN={LEN} PDU-ID={PPDU} DATA=<{DATA}>
```

Code	Value	Description
{D}		Transmission direction
	<<	Send direction
	>>	Receive direction
{PID}	1..255	Partner ID
{FU}	1..7	Functional Unit
{LEN}	1..32767	Length of the telegram
{PPDU}		PPDU type
	ConnectReq	Connect Request
	ConnectRsp	Connect Response
	ReleaseReq	Release Request
	ReleaseRsp	Release Response
	DataReq	Data Request
{ADDR}		Elcom address ¹
	I	Initiator address
	A	Acceptor address
{REASON}	0..127	Reason code
{RESULT}	0..127	Result code
{DATA}	00..FF	Data field in hex presentation

¹The following syntax is used for Elcom addresses:

```
AF={AF} PORT={PORT} IP={IP} SUFFIX={SUFFIX}
```

Code	Value	Description
{AF}	AF_INET	Address family
{PORT}	1..65535	TCP port number
{IP}	e.g. 172.16.4.1	IP address
{SUFFIX}	AA..AG	A-suffix
	BA..BG	

Logging level 4

The logging level 4 displays the APDUs (Application Protocol Data Units) human-readable presentation. The following syntax is used:

```
(4): {D} [{PID}/{FU}] LEN={LEN} APDU-ID={APDU} VERSION={VERSION} CLASS={CLASS} RESULT={RESULT}
DATA={DATA}
(4): {D} [{PID}/{FU}] APDU-ID={APDU} GTYPE={GTYPE} GNR={GNR} PERSIST={PERSIST} STATIC={STATIC}
PRIO={PRIO} GSIZE={GSIZE} OBJLEN={OBJLEN} FUNC={FUNC}
(4): {D} [{PID}/{FU}] APDU-ID={APDU} GTYPE={GTYPE} GNR={GNR} FUNC={FUNC} CF={CF} RESULT={RESULT}
(4): {D} [{PID}/{FU}] APDU-ID={APDU} GTYPE={GTYPE} GNR={GNR} INDEX1={INDEX1} INDEX2={INDEX2}
OBJID.<{N}> = "{OBJID}"
(4): {D} [{PID}/{FU}] APDU-ID={APDU} GTYPE={GTYPE} GNR={GNR} INDEX1={INDEX1} INDEX2={INDEX2} CF={CF}
RESULT={RESULT}
(4): {D} [{PID}/{FU}] APDU-ID={APDU} GTYPE={GTYPE} GNR={GNR} INDEX1={INDEX1} INDEX2={INDEX2} T0={T0}
DT={DT} TUNIT={DT} PERIODES={PERIODES}
(4): {D} [{PID}/{FU}] APDU-ID={APDU} GTYPE={GTYPE} GNR={GNR} INDEX1={INDEX1} INDEX2={INDEX2} T={T}
TRANSMOD={TRANSMOD} MORE-D={MORE} RESULT={RESULT}
(4): {D} [{PID}/{FU}] APDU-ID={APDU} GTYPE={GTYPE} GNR={GNR} INDEX1={INDEX1} INDEX2={INDEX2} T={T}
TMODE={TMODE} CTYPE={CTYPE} RESULT={RESULT}
```

Code	Value	Description
{D}		Transmission direction
	<<	Send direction
	>>	Receive direction
{PID}	1..255	Partner ID
{FU}	1..7	Functional Unit
{LEN}	0..255	Length of the user data
{APDU}		APDU type
	InitTransfer	Initiate Transfer
	SendData	Send Data
	ConfirmData	Confirm Data
	TestConnReq	Test Connection Request
	TestConnRsp	Test Connection Response
	GroupMgmtReq	Group Management Request
	GroupMgmtRsp	Group Management Response
	DefineGroupReq	Define Group Request
	DefineGroupRsp	Define Group Response
	GetGroupReq	Get Group Request
	GetGroupRsp	Get Group Response
	SpontMgmtReq	Spontaneous Management Request
	SpontMgmtRsp	Spontaneous Management Response
	ErrorPDU	Error PDU
	ConnectReq	Connect Request
	ConnectRsp	Connect Response
	CmdTransferReq	Command Transfer Request
	CmdTransferRsp	Command Transfer Response
	SendMixedDataReq	Send Mixed Data Request
SendMixedDataErrorReq	Send Mixed Data Error Request	
{VERSION}	1	Elcom version
{CLASS}	3	Elcom class
{GTYPE}	1..8	Group type
{GNR}	1..32767	Group number
{PERSIST}	0..1	Persistent group
{STATIC}	0..1	Static group
{PRIO}	0..15	Priority
{GSIZE}	1..255	Group size

Elcom90Initiator – General

{OBJLEN}	1..255	Length of object identifiers
{FUNC}	1..4	Function key
{CF}	00..FF	Control field (timestamp)
{INDEX1}	1..32767	Starting Object Index
{INDEX2}	1..32767	Ending Object Index
{N}	0..N	Element number in object identifier string
{OBJID}	[-A-Za-z0-9_]{1,255}	Object Identifier
{T0}	00..FF	Point of time for oldest group incarnation
{DT}	1..255	Time-slice between two consecutive group incarnations
{TUNIT}	1..7	Time-unit for Dt
{PERIODES}	1..32767	Number of group incarnations requested
{T}	00..FF	Time stamp
{TRANSMOD}	1..2	Mode of transmission
{MORE}	0..1	More data will follow
{TMODE}	0..3	Time mode (interpretation of T)
{CTYPE}	1..6, 252	Command type
{RESULT}	0..127	Result code
{DATA}	00..FF	User data field in hex presentation

Example of a logfile

```

24.10.13 16:29:21 Waiting for startup in active mode ...
24.10.13 16:29:21 Node connection established !
24.10.13 16:29:31 [1/7] trying remote IP 172.16.4.1:5997
24.10.13 16:29:31 [1/7] trying to bind the port to IP 0.0.0.0
24.10.13 16:29:31 [1/7] succeeded !
24.10.13 16:29:31 [1/1] trying remote IP 172.16.4.1:5997
24.10.13 16:29:31 [1/1] trying to bind the port to IP 0.0.0.0
24.10.13 16:29:31 [1/1] succeeded !
24.10.13 16:29:31 [1/2] trying remote IP 172.16.4.1:5997
24.10.13 16:29:31 [1/2] trying to bind the port to IP 0.0.0.0
24.10.13 16:29:31 [1/2] succeeded !
24.10.13 16:29:31 [1/3] trying remote IP 172.16.4.1:5997
24.10.13 16:29:31 [1/3] trying to bind the port to IP 0.0.0.0
24.10.13 16:29:31 [1/3] succeeded !
24.10.13 16:29:31 [1/5] trying remote IP 172.16.4.1:5997
24.10.13 16:29:31 [1/5] trying to bind the port to IP 0.0.0.0
24.10.13 16:29:31 [1/5] succeeded !
24.10.13 16:29:31 [1/1] connection established !
(4): <<31.107 [1/1] LEN=4 APDU-ID=ConnectReq VERSION=1 CLASS=3 DATA=
24.10.13 16:29:31 [1/2] connection established !
(4): <<31.107 [1/2] LEN=4 APDU-ID=ConnectReq VERSION=1 CLASS=3 DATA=
24.10.13 16:29:31 [1/7] connection established !
(4): <<31.107 [1/7] LEN=4 APDU-ID=ConnectReq VERSION=1 CLASS=3 DATA=
(2): <<31.107 [1/1] LEN=48 PPDU-ID=ConnectReq I-ADDR= A-ADDR= DATA=
(2): <<31.107 [1/2] LEN=48 PPDU-ID=ConnectReq I-ADDR= A-ADDR= DATA=
(2): <<31.108 [1/7] LEN=48 PPDU-ID=ConnectReq I-ADDR= A-ADDR= DATA=
24.10.13 16:29:31 [1/3] connection established !
(4): <<31.108 [1/3] LEN=4 APDU-ID=ConnectReq VERSION=1 CLASS=3 DATA=
24.10.13 16:29:31 [1/5] connection established !
(4): <<31.108 [1/5] LEN=4 APDU-ID=ConnectReq VERSION=1 CLASS=3 DATA=
(2): <<31.108 [1/3] LEN=48 PPDU-ID=ConnectReq I-ADDR= A-ADDR= DATA=
(2): <<31.108 [1/5] LEN=48 PPDU-ID=ConnectReq I-ADDR= A-ADDR= DATA=
(2): >> 31.109 [1/1] LEN=49 PPDU-ID=ConnectRsp I-ADDR= A-ADDR= RESULT=0 DATA=
(4): >> 31.109 [1/1] LEN=4 APDU-ID=ConnectRsp VERSION=1 CLASS=3 RESULT=0 DATA=
(2): >> 31.110 [1/2] LEN=49 PPDU-ID=ConnectRsp I-ADDR= A-ADDR= RESULT=0 DATA=
(4): >> 31.110 [1/2] LEN=4 APDU-ID=ConnectRsp VERSION=1 CLASS=3 RESULT=0 DATA=
(2): >> 31.110 [1/7] LEN=49 PPDU-ID=ConnectRsp I-ADDR= A-ADDR= RESULT=0 DATA=
(4): >> 31.110 [1/7] LEN=4 APDU-ID=ConnectRsp VERSION=1 CLASS=3 RESULT=0 DATA=
(2): >> 31.111 [1/3] LEN=49 PPDU-ID=ConnectRsp I-ADDR= A-ADDR= RESULT=0 DATA=
(4): >> 31.111 [1/3] LEN=4 APDU-ID=ConnectRsp VERSION=1 CLASS=3 RESULT=0 DATA=
(2): >> 31.112 [1/5] LEN=49 PPDU-ID=ConnectRsp I-ADDR= A-ADDR= RESULT=0 DATA=
(4): >> 31.112 [1/5] LEN=4 APDU-ID=ConnectRsp VERSION=1 CLASS=3 RESULT=0 DATA=
24.10.13 16:29:31 Connection to partner 1 established !
(4): <<31.112 [1/1] APDU-ID=GroupMgntReq GTYPE=0 GNR=0 PERSIST=0 STATIC=0 PRIO=0 GSIZE=0 OBJLEN=0 FUNC=4
(2): <<31.112 [1/1] LEN=9 PPDU-ID=DataReq DATA=
(2): >> 31.112 [1/1] LEN=19 PPDU-ID=DataReq DATA=
(4): >> 31.112 [1/1] APDU-ID=GroupMgntRsp GTYPE=0 GNR=0 FUNC=4 CF= RESULT=0
(4): <<31.113 [1/1] APDU-ID=GroupMgntReq GTYPE=2 GNR=1 PERSIST=0 STATIC=0 PRIO=0 GSIZE=255 OBJLEN=255 FUNC=1
(2): <<31.113 [1/1] LEN=9 PPDU-ID=DataReq DATA=
(2): >> 31.113 [1/1] LEN=19 PPDU-ID=DataReq DATA=
(4): >> 31.113 [1/1] APDU-ID=GroupMgntRsp GTYPE=2 GNR=1 FUNC=1 CF= RESULT=0
(4): <<31.113 [1/1] APDU-ID=GroupMgntReq GTYPE=1 GNR=2 PERSIST=0 STATIC=0 PRIO=1 GSIZE=255 OBJLEN=255 FUNC=1
(2): <<31.113 [1/1] LEN=9 PPDU-ID=DataReq DATA=
(2): >> 31.114 [1/1] LEN=19 PPDU-ID=DataReq DATA=
(4): >> 31.114 [1/1] APDU-ID=GroupMgntRsp GTYPE=1 GNR=2 FUNC=1 CF= RESULT=0
(4): <<31.114 [1/1] APDU-ID=GroupMgntReq GTYPE=3 GNR=3 PERSIST=0 STATIC=0 PRIO=1 GSIZE=255 OBJLEN=255 FUNC=1
(2): <<31.114 [1/1] LEN=9 PPDU-ID=DataReq DATA=
(2): >> 31.114 [1/1] LEN=19 PPDU-ID=DataReq DATA=
(4): >> 31.114 [1/1] APDU-ID=GroupMgntRsp GTYPE=3 GNR=3 FUNC=1 CF= RESULT=0
(4): <<31.115 [1/1] APDU-ID=GroupMgntReq GTYPE=6 GNR=4 PERSIST=0 STATIC=0 PRIO=0 GSIZE=255 OBJLEN=255 FUNC=1
(2): <<31.115 [1/1] LEN=9 PPDU-ID=DataReq DATA=
(2): >> 31.115 [1/1] LEN=19 PPDU-ID=DataReq DATA=
(4): >> 31.115 [1/1] APDU-ID=GroupMgntRsp GTYPE=6 GNR=4 FUNC=1 CF= RESULT=0
(4): <<31.115 [1/1] APDU-ID=GroupMgntReq GTYPE=7 GNR=5 PERSIST=0 STATIC=0 PRIO=0 GSIZE=255 OBJLEN=255 FUNC=1
(2): <<31.115 [1/1] LEN=9 PPDU-ID=DataReq DATA=
(2): >> 31.116 [1/1] LEN=19 PPDU-ID=DataReq DATA=
(4): >> 31.116 [1/1] APDU-ID=GroupMgntRsp GTYPE=7 GNR=5 FUNC=1 CF= RESULT=0
(4): <<31.116 [1/1] APDU-ID=GroupMgntReq GTYPE=5 GNR=6 PERSIST=0 STATIC=0 PRIO=0 GSIZE=255 OBJLEN=255 FUNC=1
(2): <<31.116 [1/1] LEN=9 PPDU-ID=DataReq DATA=
(2): >> 31.116 [1/1] LEN=19 PPDU-ID=DataReq DATA=
(4): >> 31.116 [1/1] APDU-ID=GroupMgntRsp GTYPE=5 GNR=6 FUNC=1 CF= RESULT=0
(4): <<31.117 [1/1] APDU-ID=DefineGroupReq GTYPE=2 GNR=1 INDEX1=1 INDEX2=1
OBJID.(0) = "Breaker_X5";
(2): <<31.117 [1/1] LEN=21 PPDU-ID=DataReq DATA=
(2): >> 31.117 [1/1] LEN=23 PPDU-ID=DataReq DATA=
(4): >> 31.117 [1/1] APDU-ID=DefineGroupRsp GTYPE=2 GNR=1 INDEX1=1 INDEX2=1 CF= RESULT=
(4): <<31.117 [1/1] APDU-ID=DefineGroupReq GTYPE=1 GNR=2 INDEX1=1 INDEX2=2
OBJID.(0) = "Current_I2";

```

Elcom90Initiator – General

```

OBJID.(1)          = "Voltage_U1";
(2): <<31.117 [1/1] LEN=32 PPDU-ID=DataReq DATA=
(2): >> 31.118 [1/1] LEN=24 PPDU-ID=DataReq DATA=
(4): >> 31.118 [1/1] APDU-ID=DefineGroupRsp GTYPE=1 GNR=2 INDEX1=1 INDEX2=2 CF= RESULT=
(4): <<31.118 [1/1] APDU-ID=DefineGroupReq GTYPE=3 GNR=3 INDEX1=1 INDEX2=1
      OBJID.(0)          = "TAP_POS_T43";
(2): <<31.118 [1/1] LEN=22 PPDU-ID=DataReq DATA=
(2): >> 31.118 [1/1] LEN=23 PPDU-ID=DataReq DATA=
(4): >> 31.118 [1/1] APDU-ID=DefineGroupRsp GTYPE=3 GNR=3 INDEX1=1 INDEX2=1 CF= RESULT=
(4): <<31.118 [1/2] APDU-ID=SpontMgntReq GTYPE=2 GNR=1 FUNC=1
(2): <<31.119 [1/2] LEN=6 PPDU-ID=DataReq DATA=
(2): >> 31.119 [1/2] LEN=7 PPDU-ID=DataReq DATA=
(4): >> 31.119 [1/2] APDU-ID=SpontMgntRsp GTYPE=2 GNR=1 FUNC=1 RESULT=0
(4): <<31.119 [1/3] APDU-ID=SpontMgntReq GTYPE=1 GNR=2 FUNC=1
(2): <<31.119 [1/3] LEN=6 PPDU-ID=DataReq DATA=
(2): >> 31.120 [1/3] LEN=7 PPDU-ID=DataReq DATA=
(4): >> 31.120 [1/3] APDU-ID=SpontMgntRsp GTYPE=1 GNR=2 FUNC=1 RESULT=0
(4): <<31.120 [1/3] APDU-ID=SpontMgntReq GTYPE=3 GNR=3 FUNC=1
(2): <<31.120 [1/3] LEN=6 PPDU-ID=DataReq DATA=
(2): >> 31.120 [1/3] LEN=7 PPDU-ID=DataReq DATA=
(4): >> 31.120 [1/3] APDU-ID=SpontMgntRsp GTYPE=3 GNR=3 FUNC=1 RESULT=0
(4): <<31.121 [1/5] APDU-ID=InitTransfer GTYPE=2 GNR=1 INDEX1=0 INDEX2=0 T0= DT=1 TUNIT=6 PERIODES=1
(2): <<31.121 [1/5] LEN=21 PPDU-ID=DataReq DATA=
(2): >> 31.121 [1/5] LEN=21 PPDU-ID=DataReq DATA=
(4): >> 31.121 [1/5] APDU-ID=SendData GTYPE=2 GNR=1 INDEX1=1 INDEX2=1 T= TRANSMOD=1 MORE-D=0 RESULT=0 DATA=
(4): <<31.121 [1/5] APDU-ID=ConfirmData GTYPE=2 GNR=1 TRANSMOD=1 RESULT=0
(2): <<31.121 [1/5] LEN=7 PPDU-ID=DataReq DATA=
(4): <<31.121 [1/5] APDU-ID=InitTransfer GTYPE=1 GNR=2 INDEX1=0 INDEX2=0 T0= DT=1 TUNIT=6 PERIODES=1
(2): <<31.121 [1/5] LEN=21 PPDU-ID=DataReq DATA=
(2): >> 31.122 [1/5] LEN=30 PPDU-ID=DataReq DATA=
(4): >> 31.122 [1/5] APDU-ID=SendData GTYPE=1 GNR=2 INDEX1=1 INDEX2=2 T= TRANSMOD=1 MORE-D=0 RESULT=0 DATA=
(4): <<31.122 [1/5] APDU-ID=ConfirmData GTYPE=1 GNR=2 TRANSMOD=1 RESULT=0
(2): <<31.122 [1/5] LEN=7 PPDU-ID=DataReq DATA=
(4): <<31.122 [1/5] APDU-ID=InitTransfer GTYPE=3 GNR=3 INDEX1=0 INDEX2=0 T0= DT=1 TUNIT=6 PERIODES=1
(2): <<31.122 [1/5] LEN=21 PPDU-ID=DataReq DATA=
(2): >> 31.123 [1/5] LEN=23 PPDU-ID=DataReq DATA=
(4): >> 31.123 [1/5] APDU-ID=SendData GTYPE=3 GNR=3 INDEX1=1 INDEX2=1 T= TRANSMOD=1 MORE-D=0 RESULT=0 DATA=
24.10.13 16:29:31 Startup successfully completed !
(4): <<31.123 [1/5] APDU-ID=ConfirmData GTYPE=3 GNR=3 TRANSMOD=1 RESULT=0
(2): <<31.123 [1/5] LEN=7 PPDU-ID=DataReq DATA=

```

Elcom90Initiator – Parameter

.name

default: ELC90I

Module name

Partner List

.partner.{{PID}} {{PID}}

default: 1 minimum: 1 maximum: 255

Partner ID

Any unique number to identify the connection to a specific communication partner.

Communication parameters

.partner.{{PID}}.client_ip

pattern: ^[0-9]+\.[0-9]+\.[0-9]+\.[0-9]+\$

Low level client IP address (optional)

The IP address used by this client for communication on OSI layer 3 can be specified here.

.partner.{{PID}}.client_port

minimum: 1 maximum: 65535

Low level client TCP/IP port (optional)

The TCP/IP port used by this client for communication on OSI layer 4 can be specified here.

.partner.{{PID}}.server_ip

pattern: ^[0-9]+\.[0-9]+\.[0-9]+\.[0-9]+\$

Server IP address

This value defines the server IP address to which the connection is to be opened.

.partner.{{PID}}.server_port

default: 5997 minimum: 1 maximum: 65535

Server TCP/IP port

This defines the TCP/IP port used by the server.

User element parameters

.partner.{{PID}}.initiator_ip

pattern: ^[0-9]+\.[0-9]+\.[0-9]+\.[0-9]+\$

High level Initiator IP address

This parameter defines the IP address of the Initiator. It is required to identify an Elcom-90 Initiator in the Responder part (on OSI layer 6).

.partner.{{PID}}.initiator_port

default: 1030 minimum: 1 maximum: 65535

High level Initiator TCP/IP port

This defines the TCP/IP port used by the Initiator. It is required to identify an Elcom-90 Initiator in the Responder part (on OSI layer 6).

Group List

.partner.{{PID}}.group.{{GNR}} {{GNR}}

default: 1 minimum: 1 maximum: 32767

Group number

Group settings

.partner.{{PID}}.group.{{GNR}}.gtype = { Measure | Status | Discrete | BinaryCommand | AnalogSetpoint | DigitalSetpoint }

default: Status

Group type

TYPE	DESCRIPTION
Measure	Floating-point values
Status	3-state values (on/off/between)
Discrete	Two octet integer
BinaryCommand	2-state commands (on/off)
AnalogSetpoint	Floating point values
DigitalSetpoint	Two octet integer command values

.partner.{{PID}}.group.{{GNR}}.gtype = Measure

.partner.{{PID}}.group.{{GNR}}.cyclic = { TRUE | FALSE }

default: FALSE

Enable cyclic transmission (=TRUE) or not (=FALSE)

If this parameter is set to "TRUE", data is transferred cyclically by the Periodic Data Transfer FU. If this parameter is set to "FALSE", data is transferred spontaneously by the Unsolicited Data Transfer FU.

.partner.{{PID}}.group.{{GNR}}.gtype = Status

.partner.{{PID}}.group.{{GNR}}.cyclic = { TRUE | FALSE }

default: FALSE

Enable cyclic transmission (=TRUE) or not (=FALSE)

If this parameter is set to "TRUE", data is transferred cyclically by the Periodic Data Transfer FU. If this parameter is set to "FALSE", data is transferred spontaneously by the Unsolicited Data Transfer FU.

.partner.{{PID}}.group.{{GNR}}.gtype = Discrete

.partner.{{PID}}.group.{{GNR}}.cyclic = { TRUE | FALSE }

default: FALSE

Enable cyclic transmission (=TRUE) or not (=FALSE)

If this parameter is set to "TRUE", data is transferred cyclically by the Periodic Data Transfer FU. If this parameter is set to "FALSE", data is transferred spontaneously by the Unsolicited Data Transfer FU.

.partner.{{PID}}.group.{{GNR}}.gtype = BinaryCommand

.partner.{{PID}}.group.{{GNR}}.sbo = { TRUE | FALSE }

default: FALSE

Select commands before execution (=TRUE) or not (=FALSE)

This parameter determines whether a command is selected before execution (=TRUE) or executed directly (=FALSE).

.partner.{{PID}}.group.{{GNR}}.gtype = AnalogSetpoint

.partner.{{PID}}.group.{{GNR}}.sbo = { TRUE | FALSE }

default: FALSE

Select commands before execution (=TRUE) or not (=FALSE)

This parameter determines whether a command is selected before execution (=TRUE) or executed directly (=FALSE).

.partner.{{PID}}.group.{{GNR}}.gtype = DigitalSetpoint

.partner.{{PID}}.group.{{GNR}}.sbo = { TRUE | FALSE }

default: FALSE

Select commands before execution (=TRUE) or not (=FALSE)

This parameter determines whether a command is selected before execution (=TRUE) or executed directly (=FALSE).

.partner.{{PID}}.group.{{GNR}}.prio

default: 0 minimum: 0 maximum: 15

Priority class

The smaller the value specified here, the higher is the priority of the group. The valid range is from 0 to 15. In case of spontaneous transmission precedence is given to groups with a higher priority. In case of cyclic transmission, the following rule applies: The smaller the value specified here, the shorter the cycle time is. However, the priority mechanism is not supported by all responders.

.partner.{{PID}}.group.{{GNR}}.gsize

default: 20 minimum: 1 maximum: 255

Group size

The maximum number of object within this group can be defined here. The valid range is from 1 to 255.

.partner.{{PID}}.group.{{GNR}}.objlen

default: 12 minimum: 1 maximum: 255

Object identifier length

The maximum number of characters that may be used for the names of the objects within this group can be defined here. The valid range is from 1 to 255.

Object List

.partner.{{PID}}.object.{{OBJID}}

{OBJID}

pattern: ^[-A-Za-z0-9_]{1,255}\$

Object Identifier

Group List

.partner.{{PID}}.object.{{OBJID}}.group.[group]

group

default: 1

Transmission groups

Imported from table

.partner.{{PID}}.import

.partner.{{PID}}.import.object for details see [.partner.{{PID}}.object](#)

General parameters

.keepalive

unit: [s] default: 5 minimum: 1 maximum: 20

TCP keepalive interval

.timeout

unit: [s] default: 10 minimum: 1 maximum: 65535

Request Timeout

If no response to a request is received within the period of time specified here, the connection to the corresponding communication partner is considered as to be disturbed. All subconnections will be aborted and reestablished.

.utc_time = { TRUE | FALSE }

default: TRUE

use UTC timezone (=TRUE) or local timezone (=FALSE) for time conversion

If this parameter is set on 'TRUE', timestamps within sent and received telegrams are treated as UTC times. If set on 'FALSE', timestamps are treated as to be local times.

.utc_time = FALSE

.tzsource = { OS | SYSTEM | SPECIFIC | MANUAL }

default: OS

type of timezone conversion mechanism

Defines, which timezone calculation mechanism is used by this protocol stack for conversion of local times into UTC times and vice versa. Following settings may be applied:

VALUE	DESCRIPTION
OS	use operating system builtin time conversion functions, based on global timezone setting
SYSTEM	use ipConv builtin time conversion functions, based on global timezone setting
SPECIFIC	timezone region and zone explicitly used by this protocol stack
MANUAL	timezone string in Posix notation explicitly used by this protocol stack

.tzsource = SPECIFIC

.tzregion = {...}

default: Etc

Region

.tzregion = Africa

.tzzone = {...}

default: Abidjan

Zone

.tzregion = America/Kentucky

.tzzone = { Louisville | Monticello }

default: Louisville

Zone

.tzregion = America/Indiana

.tzzone = { Indianapolis | Knox | Marengo | Petersburg | Tell_City | Vevay | Vincennes | Winamac }

default: Indianapolis

Zone

.tzregion = America/North_Dakota

.tzzone = { Beulah | Center | New_Salem }

default: Center

Zone

.tzregion = America/Argentina

.tzzone = { Buenos_Aires | Catamarca | ComodRivadavia | Cordoba | Jujuy | La_Rioja | Mendoza | Rio_Gallegos | Salta | San_Juan | San_Luis | Tucuman | Ushuaia }

default: Buenos_Aires

Zone

.tzregion = America

.tzone = {...}
default: Adak
Zone

.tzregion = Antarctica

**.tzone = { Casey | Davis | DumontDUrville | Macquarie | Mawson |
McMurdo | Palmer | Rothera | South_Pole | Syowa | Vostok }**
default: Casey
Zone

.tzregion = Arctic

.tzone = { Longyearbyen }
default: Longyearbyen
Zone

.tzregion = Asia

.tzone = {...}
default: Aden
Zone

.tzregion = Atlantic

**.tzone = { Azores | Bermuda | Canary | Cape_Verde | Faeroe | Faroe |
Jan_Mayen | Madeira | Reykjavik | South_Georgia | St_Helena |
Stanley }**
default: Azores
Zone

.tzregion = Australia

**.tzone = { ACT | Adelaide | Brisbane | Broken_Hill | Canberra |
Currie | Darwin | Eucla | Hobart | LHI | Lindeman | Lord_Howe |
Melbourne | North | NSW | Perth | Queensland | South | Sydney |
Tasmania | Victoria | West | Yancowinna }**
default: ACT
Zone

.tzregion = Brazil

.tzone = { Acre | DeNoronha | East | West }
default: Acre
Zone

.tzregion = Canada

**.tzone = { Atlantic | Central | Eastern | East-Saskatchewan |
Mountain | Newfoundland | Pacific | Saskatchewan | Yukon }**
default: Atlantic
Zone

.tzregion = Chile

.tzone = { Continental | EasterIsland }
default: Continental
Zone

.tzregion = Etc

.tzone = { GMT | GMT0 | GMT-0 | GMT+0 | GMT-1 | GMT+1 | GMT-10 | GMT+10 | GMT-11 | GMT+11 | GMT-12 | GMT+12 | GMT-13 | GMT-14 | GMT-2 | GMT+2 | GMT-3 | GMT+3 | GMT-4 | GMT+4 | GMT-5 | GMT+5 | GMT-6 | GMT+6 | GMT-7 | GMT+7 | GMT-8 | GMT+8 | GMT-9 | GMT+9 | Greenwich | UCT | Universal | UTC | Zulu }

default: UTC

Zone

.tzregion = Europe

.tzone = { ... }

default: Berlin

Zone

.tzregion = Indian

.tzone = { Antananarivo | Chagos | Christmas | Cocos | Comoro | Kerguelen | Mahe | Maldives | Mauritius | Mayotte | Reunion }

default: Antananarivo

Zone

.tzregion = Mexico

.tzone = { BajaNorte | BajaSur | General }

default: BajaNorte

Zone

.tzregion = Pacific

.tzone = { Apia | Auckland | Chatham | Chuuk | Easter | Efate | Enderbury | Fakaofu | Fiji | Funafuti | Galapagos | Gambier | Guadalcanal | Guam | Honolulu | Johnston | Kiritimati | Kosrae | Kwajalein | Majuro | Marquesas | Midway | Nauru | Niue | Norfolk | Noumea | Pago_Pago | Palau | Pitcairn | Pohnpei | Ponape | Port_Moresby | Rarotonga | Saipan | Samoa | Tahiti | Tarawa | Tongatapu | Truk | Wake | Wallis | Yap }

default: Apia

Zone

.tzregion = US

.tzone = { Alaska | Aleutian | Arizona | Central | Eastern | East-Indiana | Hawaii | Indiana-Starke | Michigan | Mountain | Pacific | Pacific-New | Samoa }

default: Alaska

Zone

.tzsource = MANUAL

.tzstring

default: CET-1CEST,M3.5.0/2,M10.5.0/3

Timezone string in Posix notation

The value of the TZ string can be in one of two formats. The first format is used when there is no Daylight Saving Time (or summer time) in the local time zone:

std offset

The *std* string specifies the name of the time zone. It must be three or more characters long and must not contain a leading colon, embedded digits, commas, nor plus and minus signs. There is no space character separating the time zone name from the *offset*, so these restrictions are necessary to parse the specification correctly.

The *offset* specifies the time value you must add to the local time to get a Coordinated Universal Time value. It has syntax like `[+|-]hh[:mm[:ss]]`. This is positive if the local time zone is west of the Prime Meridian and negative if it is east. The hour must be between 0 and 23, and the minute and seconds between 0 and 59.

For example, here is how we would specify Eastern Standard Time, but without any Daylight Saving Time alternative:

```
EST+5
```

The second format is used when there is Daylight Saving Time:

```
std offset dst [offset] , start[ / time ] , end[ / time ]
```

The initial *std* and *offset* specify the standard time zone, as described above. The *dst* string and *offset* specify the name and offset for the corresponding Daylight Saving Time zone; if the *offset* is omitted, it defaults to one hour ahead of standard time.

The remainder of the specification describes when Daylight Saving Time is in effect. The *start* field is when Daylight Saving Time goes into effect and the *end* field is when the change is made back to standard time. The following formats are recognized for these fields:

Jn

This specifies the Julian day, with *n* between 1 and 365. February 29 is never counted, even in leap years.

n

This specifies the Julian day, with *n* between 0 and 365. February 29 is counted in leap years.

Mm . w . d

This specifies day *d* of week *w* of month *m*. The day *d* must be between 0 (Sunday) and 6. The week *w* must be between 1 and 5; week 1 is the first week in which day *d* occurs, and week 5 specifies the *last d* day in the month. The month *m* should be between 1 and 12.

The *time* fields specify when, in the local time currently in effect, the change to the other time occurs. If omitted, the default is 02:00:00.

For example, here is how you would specify the Eastern time zone in the United States, including the appropriate Daylight Saving Time and its dates of applicability. The normal offset from UTC is 5 hours; since this is west of the prime meridian, the sign is positive. Summer time begins on the first Sunday in April at 2:00am, and ends on the last Sunday in October at 2:00am.

```
EST+5EDT,M4.1.0/2,M10.5.0/2
```

The schedule of Daylight Saving Time in any particular jurisdiction has changed over the years. To be strictly correct, the conversion of dates and times in the past should be based on the schedule that was in effect then. However, this format has no facilities to let you specify how the schedule has changed from year to year. The most you can do is specify one particular schedule usually the present day schedule and this is used to convert any date, no matter when.

(Source: <http://www.gnu.org>, TZ Variable, 26.02.2013)

.sel_timeout

unit: [s] default: 30 minimum: 1 maximum: 65535

timeout for command selection

Defines the command selection monitoring timeout in seconds. If no command with the qualifier set to execute or cancel is received from the Node within the time specified here, the command is cancelled.

.exec_timeout

unit: [s] default: 30 minimum: 1 maximum: 65535

timeout for command execution

Elcom90Initiator – General

Defines the command execution monitoring timeout in seconds. The initiator can send CBXC (check back before execute command), EXC (execute command), IHC (inhibit command) or IXC (immediate execute command) requests to the responder. If no reply is received from the responder within the time specified here, the command is cancelled.

.startup_time

unit: [s] default: 60 minimum: 1 maximum: 600

Startup timeout

Defines the startup monitoring timeout in seconds. In case of normal operation the module sends the internal message "int-startup" with the "APP" value to the node, when database is completely updated. That is the case after receiving all information from all configured RTUs. If the module does not receive all expected telegrams within the configured timeout, and therefore database cannot be completed, nevertheless the internal startup message is transmitted to the node, to enable the startup of the higher-order protocol modules.

.active = { TRUE | FALSE }

default: TRUE

startup in active mode (=TRUE) or in passive mode (=FALSE)

This parameter defines the state, in which the module starts up it's operation in general. If set on "TRUE", the module starts up in active state. If set on "FALSE", the module starts up in passive state and can be activated only with the normalized information "int-active" with the "APP" value from the node.

.level = { 0 | 1 | 2 | 4 | 6 | 7 }

default: 0

Logging Level

LEVEL	DESCRIPTION
0	no logging
1	hexadecimal representation of transport layer telegrams
2	decoded representation of presentation layer frames
4	decoded representation of application layer frames
6	levels 2 and 4
7	levels 1, 2 and 4