Product Manual

AT Command Set for In-Premises Display

Current Firmware: IPD firmware version r300
EmberZNet 4.6.4 stack

Rev:3
February 2013

Table of Contents

1 General

1.1 Revision History

Version	Rev	Date	Change Essentials	Reason	Author
1	0	01/05/2012	Initial revision		Yuanlong Liu
1	1	01/11/2012	Review	Remove redundancies	Yuanlong Liu
1	2	20/01/2012	Review	Correct description of S0A	Yuanlong Liu
1	3	20/02/2012	Review	Added description of S61	Yuanlong Liu

1.2 Review History

Rev	Date	Reviewer	Review Results

1.3 Approval History

Version	Date	Approved by	Evidence of Approval

1.4 References

This document is developed on the basis of the following:

[1] ETRX3 Series Product Manuals - The Product Manuals concentrate on the hardware specification of the modules.

[2] Development Kit User Guides - The Development Kit User Guide contains all of the information required to set up your development kit and run firmware upgrades where necessary.

[3] Application notes from www.telegesis.com

[4] ZigBee PRO specification, document 053474r17

[5] ZigBee ZCL specification, document 075123r02ZB

[6] ZigBee Smart Energy Profile 1.1 specification Revision 16, document 075356r16ZB

[7] ZigBee Cluster Library Specification 075123r03ZB

It is highly recommended to refer to the ZigBee Cluster Library and ZigBee Smart Energy Profile specification to further understand the various AT Commands and their usage.

2 Overview

2.1 Objective

This document describes the SE IPD AT Command Set firmware to be used with Telegesis ZigBee PRO wireless meshing modules on Smart Energy (SE) In-Premises Display (IPD) devices.

SE IPD AT Command Set provides a simple AT-style command interface to implement the IPD of the ZigBee SE Application Profile. Using the AT-Command interface described in this document enables quick development of an SE IPD device, which shorten R&D period significantly. It enables developers to simply design and build ZigBee Smart Energy compliant IPD products without ever having to do embedded firmware or RF work.

The Telegesis ZigBee modules are designated based on the industry leading EmberZNet ZigBee stack. They can be built into any device and provide a low cost, low power ZigBee solution. RF experience or expertise is not required to apply this powerful wireless networking capability to your products. Telegesis ZigBee Modules offer fast integration opportunities and the shortest possible time to market for your product. This document is meant as an AT-Command and S-Register reference for SE IPD R3xx firmware based on EmberZNet4.x.

2.2 ZigBee Smart Energy Profile

The ZigBee SE Profile is an application profile which defines the messages necessary to enable ‘smart energy’ functionality between metering equipment and Home Area Network (HAN) devices. This includes such activities as reading data from energy or water meters, demand response/load control applications, communicating price signals, and presenting simple information or text messages to consumers via in-home devices.

2.2.1 Smart Energy Devices

Various types of devices can participate in a HAN to enable SE applications and market programs. Typical devices are listed below:

- **Energy Service Interface (ESI)** connects the energy supply company communication network to the metering and energy management devices within the home. It may be installed within a meter, thermostat, or In-Premises Display, or may be a standalone device, and it will contain another non-ZigBee communication module (e.g. power-line carrier, RF, GPRS, broadband Internet connection).
- **Metering devices** is a meter (electricity, gas, water, heat, etc.) that is fitted with a ZigBee device. Depending on what is being metered, the device may be capable of immediate (requested) reads or it will autonomously send readings periodically. A Metering end device may also be capable of communicating certain status indicators (e.g. battery low, tamper detected).
- **Programmable Communicating Thermostats** combine the flexibility of modern thermostats, with the ability to curtail HVAC usage to heat and cool homes in a more energy efficient manner.
- **Load Control Devices** provide direct load curtailment at a specific service outlet within the home, such as a pool pump, water heater, etc.

- **Range Extender Device** is a simple device that acts as a router for other devices.
- **Smart Appliance Device** can participate in energy management activities on the ZigBee network (e.g. Utilities initiate a demand response or pricing event or the appliance actively informs customers via in home displays of when or how energy is being used).
- **In-Premise Displays** display energy consumption, load profile over time, and pricing data to involve the consumer in an interactive dialog, which informs the consumer of energy management options and potential savings.

2.2.2 In-Premise Display

The In-Premise Display device relays energy consumption data to the user by way of a graphical or text display. The display may or may not be an interactive device. At a minimum at least one of the following is displayed: current energy usage, a history over selectable periods, pricing information, or text messages. The display may also show critical pricing information to advise the customer when peaks are due to occur so that they can take appropriate action. Table 1 lists the clusters implemented in IPD AT-command set firmware. Other optional clusters such as prepayment and Load control & demand response clusters would be included in later version.

Cluster	Implementation	IPD R1xx Support	Description
Basic	Server	Supported	Provides basic information about a device
Key Establishment	Server & Client	Supported	Provides interface for key establishment between devices for secure communication
Time	Client	Supported	Provides a basic interface to a real-time clock
Simple Metering	Client	Supported	Provides the mechanism to retrieve usage information from Electric, Gas, Water, and Thermal metering devices
Price	Client	Supported	Provides the mechanism for communicating Gas, Energy, or Water pricing information within the premise. This pricing information is distributed to the Energy Service Portal (ESI) from either the utilities or from regional energy providers
Message	Client	Supported	Provides the interface for passing text messages between ZigBee devices. Messages are expected to be delivered via the ESI and then unicast to all individually registered devices implementing this cluster
Demand Response and Load Control	Client	Supported	Provides an interface to the functionality of SE Demand Response and Load Control.

Table 1: Clusters Supported by the In-Premise Display Device

2.3 Compatibility with Other Devices

The IPD R3xx AT-Command line Interpreter is based on the ZigBee Smart Energy R16 specification [6], using the ZigBee PRO feature set [4]. Interoperability with other devices which implement the Smart Energy Application Profile and use the ZigBee PRO feature set is supported.

However, IPD R3xx is not compatible with the devices that do not implement the Smart Energy Profile. Also it is not compatible with earlier version of ZigBee, including the general AT Command Set R3xx and R2xx firmware from Telegesis.

3 AT Style Command Conventions

To simplify the communication with the Telegesis modules, an AT-style command set, similar to the industry standard Hayes modem control language, is used.

Each command must be preceded by an "AT" or "at" prefix. To terminate a command enter <CR>. Any data not following this pattern is either not accepted by the module or will cause an error message in response.

Commands are followed by an optional response that includes <CR><LF><Response><CR><LF> and/or a prompt <CR><LF><Prompt><CR><LF> where the prompt could also be an error message.

Example:

ATS00? <CR>
<CR><LF>FFFF<CR><LF>
<CR><LF>OK<CR><LF>

It is recommended to wait for an “OK” or “ERROR: XX” prompt before issuing the next command.

Any data which is prompted to the user is delivered in the format <CR><LF><prompt><CR><LF>.

Example:

<CR><LF><JPAN:11,1789,37BF1CD42CC5E673><CR><LF>

A prompt intersecting a command being entered will not affect the command itself.

Throughout this document, only the responses and prompts are presented, <CR><LF> are omitted intentionally. Sequences of AT commands in a single line are not supported.

Telegesis ETRX357 features a 256 byte FIFO buffer for incoming radio messages, which allow rapid reception of multiple messages without loss of characters.

Read Command **ATXXX?**	Commands ending with a ‘?’ return the currently set value of the parameter or parameters
Write Command **ATXXX=<…>**	This command sets user-definable parameters as indicated by the ‘=’ sign.
Execute Command **ATXXX**	This command executes routines of the module and returns parameters

Table 2: Types of AT commands

3.1 Parameters

Each parameter must be entered in the correct format for any of the AT commands to execute correctly. Optional parameters are marked with square brackets […]. Some general abbreviations:

XX	8-bit hexadecimal number. Valid characters are 0-9, a-f and A-F
XXXX	16-bit hexadecimal number. Valid characters are 0-9, a-f and A-F
n	Number from 0-9
s	Sign
b	Bit (0 or 1)
c	character
<PID>	16-bit hexadecimal PAN ID (0000 to FFFF)
<EPID>	64-bit hexadecimal extended PAN ID
<channel>	decimal channel (802.15.4 channel 11-26)
<password>	8 character password
<EUI64>	64-bit IEEE 802.15.4 address in hexadecimal
<data>	Custom Data

Table 3: Different formats of parameters

3.2 Prompt Overview

The following prompts can show up during the operation of the AT Command set for an IPD.

Prompt Overview	
OK	OK terminator
ERROR:XX	Error number XX occurred
ACK:XX	Acknowledgement for message XX was received
NACK:XX	Acknowledgement for message no XX was not received
LeftPAN	Local Node has left the PAN
LostPAN	Device has lost contact with Parent
JPAN:<channel>,<PID>,<EPID>	Local Node has joined PAN with given parameters
KESTARTED	Key establishment procedure has started
KECOMPLETE	Key establishment procedure completed successfully
KETERMINATE:<Terminate code(section 3.2.1.2)>	Receive Terminate Key Establishment Command
TERMINATEKE:<Terminate code (section 3.2.1.2)>	Send Terminate Key Establishment Command
PARTERLINKREQ	Send request to establish partner link key
REGSTARTED	Registration started
REGCOMPLETE	Registration completed
REGINPROGRESS	Registration in progress
BIND:<NodeID>,<Status>	Binding created between IPD and meter/ESI
UNBIND: <NodeID>,<Status>	Break the binding between IPD and meter/ESI
ATTR:<Attribute Value>	See +GETATR command description for information
TIME:<current time>	See +GETTIME command description for information

RESPATTR: <Cluster>,<AttrID>,<Status>,<AttrInfo>	See +READATR command description for information
WRITEATTR:<Cluster>,< AttrID >,<Status>	See +WRITEATR command description for information
DFTREP:<ClusterID>,<CommandID>,<Status>	Received default response
<xxDEV>:<Node ID>,<End Point> **Please note:** <xxDEV> - may be PRICEDEV, DRLCDEV, METERDEV or MSGDEV	ZigBee Service Discovery information. See +DISCOVER for more information
PROFILE: <NodeID>,<EP>,<EndTime>,<Status>, <ProfileInterval period>,<No of Periods Delivered>, **<Interval 1><Interval 2>...<Interval N>**	Energy consumption for profiling purposes. See +GMP command description for more information
PRICE: <NodeID>,<EP>,<Provider ID>, <Rate Label>,<Event ID>,<Current Time>,<Unit of Measure>,<Currency>,<Price Trailing Digit & Price Tier>,<No of Price Tiers & Register Tier>,<Start Time>,<Duration in Minutes>,<Price>,<Price Ratio>,<Generation Price>,<Generation Price Ratio>,<Alternate Cost Delivered >,<Alternate Cost Unit >,<Alternate Cost Trailing Digit >,<Number Of Block Thresholds >,<Price Control>	Price information received from the ESI. See +CURPRICE command description for more information
DRLCRECEIVED:<eventid>,<starttime>,<duration>,<deviceclass>,<group>,<criticality>,<coolingTempOffset>,<heatingTempOffset>,<coolingTempSetPoint>,<heatingTempSetPoint>,<avgLoadAdjustmentPercent>,<duty>,<control byte>	Load control event information received from ESI. See +GSE command description for more information.
DRLCSTART: <eventid>,<starttime>,<duration>, <deviceclass>,<group>,<criticality>,<coolingTempOffset>,<heatingTempOffset>,<coolingTempSetPoint>,<heatingTempSetPoint>,<avgLoadAdjustmentPercent>,<duty>,<control byte>	Load control event information received from ESI. See +GSE command description for more information.
DRLCCOMPLETE: <eventid>,<starttime>,<duration>,<deviceclass>,<group>,<criticality>,<coolingTempOffset>,<heatingTempOffset>,<coolingTempSetPoint>,<heatingTempSetPoint>,<avgLoadAdjustmentPercent>,<duty>,<control byte>	Load control event information received from ESI. See +GSE command description for more information.
DRLCOPTOUT: <eventid>,<starttime>,<duration>, <deviceclass>,<group>,<criticality>,<coolingTempOffset>,<heatingTempOffset>,<coolingTempSetPoint>,<heatingTempSetPoint>,<avgLoadAdjustmentPercent>,<duty>,<control byte>	Load control event information received from ESI. See +GSE command description for more information.
DRLCOPTIN: <eventid>,<starttime>,<duration>, <deviceclass>,<group>,<criticality>,<coolingTempOffset>,<heatingTempOffset>,<coolingTempSetPoint>,<heatingTempSetPoint>,<avgLoadAdjustmentPercent>,<duty>,<control byte>	Load control event information received from ESI. See +GSE command description for more information.
DRLCCANCELED: <eventid>,<starttime>,<duration>,<deviceclass>,<group>,<criticality>,<coolingTe	Load control event information received from ESI. See +GSE command description for more information.

mpOffset>,<heatingTempOffset>,<coolingTempSetPoint>,<heatingTempSetPoint>,<avgLoadAdjustmentPercent>,<duty>,<control byte>	
DRLCSUPERSEDED: <eventid>,<starttime>,<duration>,<deviceclass>,<group>,<criticality>,<coolingTempOffset>,<heatingTempOffset>,<coolingTempSetPoint>,<heatingTempSetPoint>,<avgLoadAdjustmentPercent>,<duty>,<control byte>	Load control event information received from ESI. See +GSE command description for more information.
DRLCEVENT: <eventid>,<starttime>,<duration>,<deviceclass>,<group>,<criticality>,<coolingTempOffset>,<heatingTempOffset>,<coolingTempSetPoint>,<heatingTempSetPoint>,<avgLoadAdjustmentPercent>,<duty>,<control byte>	Load control event information received from ESI. See +GSE command description for more information.
MESSAGE:<MessageID>,<MessageControl>,<StartTime>,<Duration in Minutes>,<Message>	Message received from the ESI. See +LASTMSG command description and Table 7 for more information
MSGCANCLE: <MessageID>,<MessageControl>	Message cancelled by ESI or Message duration expired

Table 4: Prompt overview

3.2.1 Notes of Prompt

3.2.1.1 REGINPROGRESS

Registration progress includes several steps and takes some time to complete. In order not to interfere registration, user is not expected to issue any at command during this progress. This prompt is set as a reminder when user issues following AT commands during registration progress.

AT+READATTR for read time cluster attribute

AT+SYNCTIME

AT+BINDMTR

AT+UNBINDMTR

AT+DISCOVER

AT+MATCHREQ

3.2.1.2 KETERMINATE and TERMINATEKE

This prompt will show terminate code, which is the cause for terminating key establishment.

01 UNKNOWN_ISSUER

02 BAD_KEY_CONFIRM

03 BAD_MESSAGE

04 NO_RESOURCES

05 UNSUPPORTED_SUITE

3.3 Device Overview

Table 5 gives an overview of the ZigBee device types mentioned in this document.

Device Types		ZigBee Naming Convention
COO	Coordinator	ZigBee Coordinator (ZC)
FFD	Router	ZigBee Router (ZR)
ZED	End Device (non sleepy)	ZigBee End Device (ZED)
SED	Sleepy End Device	
MED	Mobile Sleepy end Device	

Table 5: Device Overview

3.4 Addressing modes

Many of the AT commands take a device address as a parameter, which can usually be expressed in several different formats.

EUI64. 16 hexadecimal characters. This is flashed onto the chip at manufacture and cannot be changed by the user. It can be compared to the permanent MAC address of an IP-based device.

NodeID. 4 hexadecimal characters. This is allocated to the device when it joins the PAN and cannot be changed or preset, except that 0x0000 is always the coordinator. It is analogous to a temporary IP address.

FF. In many commands address FF represents the local device.

3.5 AT Command Overview

The following table gives a quick reference of all commands available.

Command Overview		
Module control and configuration		
ATI	**Display Product Identification Information**	
ATZ	**Software Reset**	
AT&F	**Restore Factory Defaults**	
AT+BLOAD	**Enter The Bootloader Menu**	
ATS	**S-Register Access**	
AT+TOKDUMP	**Display All S-Registers**	
AT+RAWZCL	**Construct A Raw ZCL Message And Send To Target**	
AT+RAWZDO	**Construct A Raw ZDO Message And Send To Target**	
AT+HELP	**Display All Available Commands**	
Network control and configuration		
AT+ESCAN	**Scan The Energy Of All Channels**	
AT+PANSCAN	**Scan For Active PAN’s**	
AT+JN	**Join Network**	
AT+JPAN	**Join Specific PAN**	
AT+DASSR	**Disassociate Remote Node From PAN**	**(ZDO)**
AT+DASSL	**Disassociate Local Node From PAN**	
AT+N	**Display Network Information**	
AT+NTABLE	**Display Neighbour Table**	**(ZDO)**
AT+RTABLE	**Display Routing Table**	**(ZDO)**

AT+IDREQ	Request Node's NodeID	(ZDO)
AT+EUIREQ	Request Node's EUI	(ZDO)
AT+NODEDESC	Request Node's Descriptor	(ZDO)
AT+POWERDESC	Request Node's Power Descriptor	(ZDO)
AT+ACTEPDESC	Request Node's Active Endpoint List	(ZDO)
AT+SIMPLEDESC	Request Endpoint's Simple Descriptor	(ZDO)
AT+MATCHREQ	Find Nodes Which Match A Specific Descriptor	(ZDO)
AT+ANNCE	Announce Local Device In The Network	(ZDO)
AT+ATABLE	Display Address Table	
AT+ASET	Set Address Table Entry	
AT+LBTABLE	Display Local Binding Table	
AT+BSET	Set Local Binding Table Entry	
AT+BCLR	Clear Local Binding Table Entry	
AT+BTABLE	Display Remote Binding Table	(ZDO)
AT+BIND	Create Binding on Remote Device	(ZDO)
AT+UNBIND	Delete Binding on Remote Device	(ZDO)
AT+PJOIN	Switch on "Permit Joining" Flag	
AT+CBKE	Initiate CBKE With ESI Or A Partner Device	
AT+RMKEY	Remove An Entry From The Local Link Key Table	
ZigBee Discovery and Binding Commands		
AT+DISCOVER	Discover SE Device On The HAN	
AT+FINDMTR	Finds A Meter/ESI On The HAN	
AT+CLUSDISC	Discover All Supported Clusters on A Remote Device	
AT+ATTRDISC	Discover Supported Attributes on A Remote Device	
AT+BINDMTR	Bind The IPD To A Meter/ESI	
AT+UNBINDMTR	Un Bind The IPD From A Meter/ESI	
Time Control commands		
AT+SETTIME	Set The Time On The IPD	
AT+GETTIME	Get The Time On The IPD	
AT+SYNCTIME	Sync The IPD's Clock With ESI	
ZCL Attributes commands		
AT+READATR	Read Attribute From Remote Device	
AT+READMATR	Read Manufacturer Specific Attribute From Remote Device	
AT+WRITEATR	Write Attribute To Remote Device	
AT+WRITEMATR	Write A Manufacturer Specific Attribute To Remote Device	
AT+SETATR	Set Value Of A Local ZCL Attribute	
AT+GETATR	Get Value Of A Local ZCL Attribute	
IPD specified commands		
AT+GMP	Send Get Meter Profile request	
AT+CURPRICE	Gets The Current Pricing From The ESI	
AT+SCHPRICE	Gets All The Scheduled Pricing From The ESI	
AT+PRICELBL	Gets The Label Assigned To The Price Tier 1 To 6 In Price Cluster	
AT+LASTMSG	Gets The Last Message From The ESI	
AT+ACKMSG	Used To Acknowledge A Message	
AT+RES	Report Event Status	
AT+OPT	OPT In Or Out Event	
AT+GSE	Get Scheduled Events From ESI	

Table 6: Command Overview

3.5.1 Common Command Set

3.5.1.1 Module Control & Configuration Commands

I – Display Product Identification Information	
Execute Command **ATI**	Response **Telegesis <DeviceName>** **<Firmware Revision>** **<EUI64>** **OK** **<DeviceName>** is the device type **<Firmware Revision>** is the firmware revision **<EUI64>** is the device's IEEE 802.15.4 identifier

Z – Software Reset	
Execute Command **ATZ**	Response **OK** Module performs software reset. All non-volatile S Registers keep the user defined values, if the module was part of a PAN it will remain part of it.

&F – Restore Local Device's Factory Defaults	
Execute Command **AT&F**	Response **OK** Module performs a factory reset. All non-volatile S Registers are updated with their factory defaults and the node leaves the currently joined network.

+BLOAD – Enter The Boot Loader Menu	
Execute Command **AT+BLOAD** Please note: the bootloader will run at a baud rate of 115k, no parity, and 8 data bits regardless of the current serial port settings.	Response **<entering bootloader>** The device leaves the AT command line and enters the Ember bootloader menu for downloading new firmware. A description of the bootloading process can be found in the section 7 of this document. A description of the bootloading process can be found in the Development Kit Product Manual.

S – S-Register Access	
Read Command **ATSXX[x]?**	Response **<data>** **OK** or **ERROR:<errorcode>** The module displays the contents of S-register xx or an error message, where <errorcode> represents the error code explained in section 4. All 16-bit registers can also be accessed bit by bit. In order to do this [x] may specify the bit which is to be read. The result when reading a single bit will always be 0 or 1.
Write Command **ATSXX[x]=<data>** *Notes* Some S-Registers require a password for write access. See S-Register description for details. **The default password is "password".** Some S-Registers are read-only and will return an error if you are trying to write to them	Response **OK** or **ERROR:<errorcode>** The data is written to S-register number XX and if applicable stored in non-volatile memory. The data format for each individual S Register is given in the S-Register description. **<errorcode>** please see section 4. All 16-bit registers individual bits can also be set or cleared by specifying the bit using [x] and setting it to either 0 or 1.

TOKDUMP – Display All S-Registers	
Execute Command **AT+TOKDUMP** *Notes* Only used on the local node.	Response **<data>** **OK** The module displays the contents of all local S-Registers. The data format for each individual S Register is given in the S-Register description in section 5.

+RAWZCL – Construct A Raw ZCL Message And Send To Target	
Execute Command **AT+RAWZCL:<NodeID>,<EP>,<ClusterID>,<data>** **<NodeID>** - 16 bit hexadecimal number, network address of a remote device. **<EP>** - 8 bit hexadecimal number, endpoint of	Response **OK** < Response data> or **ERROR:<errorcode>** (please see section 4). The response may vary depending on the sent

a remote device. **<ClusterID> -** 16 bit hexadecimal number which represents cluster ID, see section 0 **<data> -** a constructed ZCL command (please check ZigBee Cluster Specification and SE Specification for references)	command. Note: When constructing a Raw ZCL message, please check ZigBee ZCL Specification [5] and ZigBee SE Specification [6] for references. Incorrect construction can be transmitted but will not expect any response.

+RAWZDO – Construct A Raw ZDO Message And Send To Target	
Execute Command	Response
AT+RAWZDO:<NodeID>,<ClusterID>,<data> **<Node ID> -** 16 bit hexadecimal number, network address of a target device or FFFD for broadcast. **<Cluster ID>** - ZDO Command ID, see ZigBee Specification [4] for more information. **<data> -** constructed payload of a ZCL command (please check ZigBee Cluster Specification and SE Specification for references)	**OK** **CMD:<ClusterID>,PAYLOAD<Response data>** or **ERROR:<errorcode>** (please see section 4). The response may vary depending on sent command. Note: When constructing a Raw ZDO message, please check ZigBee Specification [4] for references. Incorrect construction can be transmitted but will not expect any response.

+HELP – Display All Available Commands	
Execute Command	Response
AT+HELP *Notes* Only used on the local node.	**Available Commands:** **<Description of all supported AT commands>** **OK**

3.5.1.2 Network Control & Configuration Commands

+ESCAN – Scan The Energy Of All Channels	
Execute Command	Response
AT+ESCAN *Use on:* All nodes *Note:* Scanning all channels can take up to 16 seconds. Channels masked out in S00 are not scanned.	**+ESCAN:** **11:XX** **…** **26:XX** **OK** or **ERROR:<errorcode>** (please see section 4). **XX** represents the average energy on the respective channel (see description in 6).

+PANSCAN – Scan For Active PANs	
Execute Command **AT+PANSCAN** *Use on:* All nodes *Note:* Scanning for active PANs can take up to 16 seconds. The node does not join any of the PANs found. Channels masked out in S00 are not scanned.	Response **+PANSCAN:<channel>,<PID>,<EPID>,XX,b** **OK** or **ERROR:<errorcode>** (please see section 4). **<channel>** represents the channel, **<PID>** the PAN ID, **<EPID>** the extended PAN ID, The node gives a list of all PANs found. **XX** the ZigBee stack profile (00 = Custom, 01 = ZigBee, 02 = ZigBee PRO) **b** indicates whether the network is allowing additional nodes to join (1 = joining permitted).

+JN – Join Network	
Execute Command **AT+JN** *Use on:* All nodes which are not part of a PAN *Note:* Joining a PAN can take up to 8 seconds, depending on the number of channels which need scanning. This command can only be executed if the local node is not part of a PAN already.	Response **OK** **JPAN:<channel>,<PID>,<EPID>** or **ERROR:<errorcode>** (please see section 4). The local node scans all channels selected in register S00 for the existence of a PAN. When finding any PAN which allows joining it will automatically join in via the remote node with the highest RSSI. In case registers S02 and S03 differ from the default value of all zeros the node will only join a PAN with the specified PAN ID and/or extended PAN ID.

+JPAN – Join Specific PAN	
Execute Command **AT+JPAN:[<channel>],[<PID>],[<EPID>]** **<channel>** - 2 digit decimal number which represents channel number. Range from 11 to 26. **<PID>** - 16 bit hexadecimal number which represents short PAN ID of ZigBee network. **<EPID>** - 64 bit hexadecimal number which represents extended PAN ID of ZigBee network. Each parameter is optional for input, but "," is required when the parameter is left blank. *Examples:* **AT+JPAN:20(only specifies channel)** **AT+JPAN:,1234,(only specifies PID)** *Note:* This command can only be executed if the local node is not part of a PAN already.	Response **OK** **JPAN:<channel>,<PID>,<EPID>** or **ERROR: <errorcode>** (please see section 4). The local node scans channel provided in <channel> parameter or if this parameter is missed all channels selected in register S00 for the existence of a PAN. When finding any PAN which allows joining it will automatically join in via the remote node with the highest RSSI. In case <PID> and <EPID> parameters are provided or if not provided registers S02 and S03 differ from the default value of all zeros the node will only join a PAN with the specified Pan ID and/or extended PAN ID.

+DASSR – Disassociate Remote Node from PAN	**(ZDO)**
Execute Command **AT+DASSR:<address>** Where <address> can be a node's EUI64, NodeID or address table index *Use on* All Devices *Note* Use with care when targeting a Coordinator. It will not be able to rejoin the PAN	Response **SEQ:XX** **OK** or **ERROR:<errorcode>** **<errorcode>** please see section 4. Instruct device to leave the PAN.
Remote Action Node leaves PAN	Prompt **LeftPAN**

+DASSL – Disassociate Local Device From PAN	
Execute Command **AT+DASSL** *Use on* All Devices *Note* Use with care on a Coordinator. It will not be able to rejoin the PAN	Response **OK** or **ERROR<errorcode>** Prompt **LeftPAN** **<errorcode>** please see section 4. Instruct local device to leave the PAN.

+N – Display Network Information	
Read Command **AT+N** *Use on* All Devices	Response **+N=<devicetype>,<channel>,<power>,<PID>,<EPID>** or **+N=NoPAN** followed by **OK** **<devicetype>** represents the node's functionality in the PAN (FFD,COO,ZED,SED,MED) **<power>** the node's output power in dBm **<channel>** the IEEE 802.15.4 radio channel (11-26) **<PID>** the node's PAN ID **<EPID>** the node's extended PAN ID.

+NTABLE – Display Neighbour Table	**(ZDO)**

Read Command
AT+NTABLE:XX,<address>
Where XX is the start index of the remote LQI table and <address> can be the remote node's EUI64, NodeID or address table entry.
Use on
FFD, COO as the target device
Note:
Also the local node can be the target of this command (e.g. use address table
	Response
	SEQ:XX
	OK
	or **ERROR<errorcode>**
	This command requests the target node to respond by listing its neighbour table starting from the requested index. Can be used to find the identity of all ZigBee devices in the network including non-Telegesis devices.
	Prompt (example)
	`NTable:<NodeID>,<errorcode>`
	`Length:03`
	`No.| Type |       EUI        |  ID  | LQI`
	`0.  |  FFD | 000D6F000015896B | BC04 | FF`

entry FF as the address)	`1. |  FFD | 000D6F00000B3E77 | 739D | FF`
	`2. |  FFD | 000D6F00000AAD11 | 75E3 | FF`
	In this example the neighbour table of the remote node with the short ID shown in <NodeID> contains three entries (hexadecimal), which are displayed. In case the table contains more than three entries it may be required to repeat this command and increase the index count until the full table is derived.
	In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode.
	After successful transmission, the sequence number of the unicast is stated using the “SEQ:XX” prompt. When acknowledged (or not) the accompanying “ACK:XX” (or “NACK:XX”) prompt is displayed.

+RTABLE – Display Routing Table	**(ZDO)**
Read Command	Response

AT+RTABLE:XX,<address>
Where XX is the start index of the remote routing table and <address> can be the remote node’s EUI64, NodeID or address table entry.
Use on
FFD, COO as the target device
Note:
Also the local node can be the target of this command (e.g. use address table entry FF as local address)
	SEQ:XX
	OK
	or **ERROR<errorcode>**
	This command requests the target node to respond by listing its routing table starting from the requested index.
	Prompt (example)
	`RTable:<NodeID>,<errorcode>`
	`Length:03`
	`No.| Dest | Next | Status`
	`0.  | 1234 | ABCD | 00`
	`1.  | 4321 | 739D | 00`
	`2.  | 0000 | 0000 | 03`
	In this example the routing table of the remote node with the short ID shown in <NodeID> contains 64 entries (hexadecimal 0x40), of which the first three are displayed. When the table contains more than the displayed entries it may be required to repeat this command and increase the index count until the full table is derived.
	The status shown is as described in table 2.128 of the ZigBee Pro Specification [4].
	In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode.
	After successful transmission, the sequence number of the unicast is stated using the “SEQ:XX” prompt. When acknowledged (or not) the accompanying “ACK:XX” (or “NACK:XX”) prompt is displayed..

+IDREQ – Request Node's NodeID	**(ZDO)**
Execute Command **AT+IDREQ:<Address>[,XX]** Where <Address> can be a node's EUI64, or address table entry and XX is an optional index number. In case an index number is provided, an extended response is requested asking the remote device to list its associated devices (ie children). Sends a broadcast to obtain the specified Device's NodeID and optionally also elements of its associated devices list. *Use on* All Devices *Note* Providing FF as an address table entry addresses the local node	Response **OK** or **ERROR:<errorcode>** **<errorcode>** represents the error code in
	Prompt **AddrResp:<errorcode>[,<NodeID>,<EUI64>]** **[nn. <NodeID>]** In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode. <EUI64> is the Remote node's EUI64 and <NodeID> is its NodeID. In case an extended response has been requested the requested NodeIDs from the associated devices list are listed as well.

+EUIREQ – Request Node's EUI64	**(ZDO)**
Execute Command **AT+EUIREQ:< Address>,<NodeID>[,XX]** Where <Address> is the EUI64, NodeID or address table entry of the node which is to be interrogated about the node with the NodeID specified in <NodeID>. XX is an optional index number. In case an index number is provided, an extended response is requested asking the remote device to list its associated devices (i.e. children). Sends a unicast to obtain the specified device's EUI64 and optionally also elements of its associated devices list (extended response). *Use on* All Devices *Note* Providing FF as an address table entry addresses the local node	Response **SEQ:XX** **OK** or **ERROR:<errorcode>** (please see section 4)
	Prompt **AddrResp:<errorcode>[,<NodeID>,<EUI64>]** In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode. <EUI64> is the Remote node's EUI64 and <NodeID> is its NodeID. In case an extended response has been requested the requested NodeIDs from the associated devices list are listed. After successful transmission, the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not) the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed.

+NODEDESC – Request Node's Descriptor	**(ZDO)**
Execute Command **AT+NODEDESC:<Address>,<NodeID>** Where <Address> is the EUI64, NodeID or Address table entry of the node which is to be interrogated about the node with the NodeID specified in <NodeID>. Sends a unicast to obtain the specified device's node descriptor. *Use on* All Devices *Note* Providing FF as an address table entry addresses the local node	Response **SEQ:XX** **OK** or **ERROR:<errorcode>** (see section 4)
	Prompt (example) **NodeDesc:<NodeID>,<errorcode>** **Type:FFD** **ComplexDesc:No** **UserDesc:No** **APSFlags:00** **FreqBand:40** **MacCap:8E** **ManufCode:1010** **MaxBufSize:52** **MaxInSize:0080** **SrvMask:0000** **MaxOutSize:0080** **DescCap:00** In case of an error an errorcode other than 00 will be displayed and the prompt will end after the errorcode. <NodeID> is the Remote node's NodeID. In addition the node descriptor is displayed. The individual fields of the Node Descriptor are described in section 2.3.2.3 of the ZigBee Pro specification [4]. After successful transmission, the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not) the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed.

+POWERDESC – Request Node's Power Descriptor	**(ZDO)**
Execute Command **AT+POWERDESC:<Address>,<NodeID>** Where <Address> is the EUI64, NodeID or Address table entry of the node which is to be interrogated about the node with the NodeID specified in <NodeID>. Sends a unicast to obtain the specified device's power descriptor. *Use on* All Devices	Response **SEQ:XX** **OK** or **ERROR:<errorcode>** (please see section 4).
	Prompt **PowerDesc:<NodeID>,<errorcode>[,<PowerDescriptor>]** In case of an error an errorcode other than 00 will be displayed and the prompt will end after the errorcode <NodeID> is the Remote node's NodeID. In

	addition the <PowerDescriptor> is displayed as a 16 bit hexadecimal number as described in section 2.3.2.4. of ZigBee Pro Specification. After successful transmission, the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not) the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed.

+ACTEPDESC – Request Node's Active Endpoint List	**(ZDO)**
Execute Command **AT+ACTEPDESC:<Address>,<NodeID>** Where <Address> is the EUI64, NodeID or Address table entry of the node which is to be interrogated about the node with the NodeID specified in <NodeID>. Sends a unicast to obtain the specified device's active endpoint list. *Use on* All Devices	Response **SEQ:XX** **OK** or **ERROR:<errorcode>** (please see section 4).
	Prompt **ActEpDesc:<NodeID>,<errorcode>[,XX,…]** <NodeID> is the Remote node's NodeID. In addition all active endpoints are listed as 8-bit hexadecimal numbers separated by commas. In case of an error an errorcode other than 00 will be displayed and the prompt will end after the errorcode After successful transmission, the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not) the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed.

+SIMPLEDESC – Request Endpoint's Simple Descriptor	**(ZDO)**
Execute Command **AT+SIMPLEDESC:<Address>,<NodeID>,<XX>** Where <Address> is the EUI64, NodeID or Address table entry of the node which is to be interrogated about the node with the NodeID specified in <NodeID> and XX is the number of the endpoint, which simple descriptor is to be read. XX should be hexadecimal number. Sends a unicast to obtain the specified device's active endpoint list. *Use on* All Devices	Response **SEQ:XX** **OK** or **ERROR:<errorcode>** **<errorcode>** please see section 4.
	Prompt **SimpleDesc:<NodeID>,<errorcode>** **EP:XX** **ProfileID:XXXX** **DeviceID:XXXXvXX** **InCluster:<Cluster List>**

	OutCluster:<Cluster List> In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode. <NodeID> is the Remote node's NodeID. In addition all active endpoints are listed as 8 bit hexadecimal numbers separated by commas. After successful transmission the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not) the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed.

+MATCHREQ – Find Nodes which Match a Specific Descriptor	**(ZDO)**
Execute Command **AT+MATCHREQ:<ProfileID>,<NumInClusters>[,<InClusterList>],<NumOutClusters>[,<OutClusterList>]** Where <ProfileID> Required profile ID of the device being searched for followed by a specification of required input and output clusters. If a remote node has a matching ProfileID and matches at least one of the specified clusters it will respond to this broadcast listing the matching endpoint(s). **<NumInClusters>** and **<NumOutClusters>** must be 2 hexadecimal digits *Example* **AT+MATCHREQ:0109,01,0000,02,0700,0701** *Use on* All Devices	Response **OK** or **ERROR:<errorcode>** **<errorcode>** please see section 4.
	Prompt **MatchDesc:<NodeID>,<errorcode>,XX,…** In case of an error an errorcode other than 00 will be displayed and the prompt will end after the errorcode. Where <NodeID> is the Remote node's NodeID. In addition all endpoints of this node matching the search criterion are listed as 8 bit hexadecimal numbers separated by commas.

+ANNCE – Announce Local Device In The Network	**(ZDO)**
Execute Command **AT+ANNCE** Send a ZigBee device announcement. Broadcast announcing the local node on the network. *Use on* All Devices	Response **OK** or **ERROR<errorcode>** **<errorcode>** please see in section 4.
Remote Action	Prompt **FFD:<EUI64>,<NodeID>**

The prompt above will be displayed on all nodes which can hear the announcement.

+ATABLE – Display Address Table

Read Command

AT+ATABLE

Use on

All Devices

Notes

The address table is volatile and its contents are lost if the device is powered down.

Response

```
No. | Active |  ID  | EUI
00  |   N    | 0000 |000D6F0000012345
(…)
OK
```

The Address Table contains nodes which can be addressed by referring to the corresponding address table entry. The “Active” column shows nodes to which a message is currently in flight.

+ASET – Set Address Table Entry

Read Command

AT+ASET:XX,<NodeID>,<EUI64>

Where XX is the entry number of the address table entry which is to be written. If the NodeID is unknown, the NodeID must be substituted with “FFFF”.

Use on

All Devices

Response

OK

or **ERROR:<errorcode>**

<errorcode> represents the error code in section 4.

+LBTABLE – Display Local Binding Table

Read Command

AT+LBTABLE

Use on

All Devices

The binding table is cleared by a reset

Response

```
No. |  Type | Active | LocalEP | ClusterID |      Addr        | RemEP
10. | Ucast |   No   |  01     |   DEAD    | 1234567887654321 |  01
11. |  MTO  |   No   |  01     |   DEAD    | E012345678876543 |  88
12. | Mcast |   No   |  01     |   DEAD    | CDAB
13. | Unused
14. | Unused
15. | Unused
16. | Unused
17. | Unused
18. | Unused
19. | Unused
```

+BSET – Set local Binding Table Entry

Read Command

AT+BSET:<type>,<LocalEP>,<ClusterID>,<DstAddress>[,<DstEP>]

Where:

Response

OK

or **ERROR:<errorcode>**

<errorcode> represents the error code

<Type> is the type of binding: *1= Unicast Binding with EUI64 and remote EP specified* *2= Many to one Binding with EUI64 and remote EP Specified* *3= Multicast Binding with Multicast ID Specified* **<LocalEP>** is the local endpoint **<ClusterID>** is the t cluster ID, Address is either the EUI64 of the target device, or a multicast ID **<DstEP>** the remote endpoint which is not specified in case of a multicast binding. The new binding is created in the next available free binding table entry. *Use on* All Devices	explained in section 4.

+BCLR – Clear local Binding Table Entry	
Read Command **AT+BCLR:XX** Where **XX** is the entry number of the binding table which is to be cleared. *Use on* All Devices	Response **OK** or **ERROR:<errorcode>** **<errorcode>** is explained in section 4.

+BTABLE – Display Binding Table	**(ZDO)**
Read Command **AT+BTABLE:XX,<address>** Where XX is the start index of the remote binding table and <address> can be the remote node's EUI64, NodeID or address/binding table entry. *Use on* All devices *Note:* Also the local node can be the target of this command (e.g. use address table entry FF as the address)	Response **SEQ:XX** **OK or ERROR<errorcode>** This command requests the target node to respond by listing its binding table starting from the requested index. The response indicates success or failure in sending this message. The acknowledgement as well as the actual response to this request will follow as asynchronous prompts.
	Example **AT+BTABLE:00,0000**

SEQ:01

OK

```
BTable:0000,00
Length:03
No. |     SrcAddr       | SrcEP | ClusterID |      DstAddr     | DstEP
00. | 000D6F000059474E |  01   |    DEAD   |1234567887654321 |  12
01. | 000D6F000059474E |  01   |    DEAD   |E012345678876543 |  E0
02. | 000D6F000059474E |  01   |    DEAD   | ABCD
ACK:01
```

In this example the neighbour table of the remote node with the short ID shown in <NodeID> contains three entries (hexadecimal), which are displayed. In case the table contains more than three entries it may be required to repeat this command and increase the index count until the full table is derived.

In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode.

+BIND – Create Binding on Remote Device	**(ZDO)**
Read Command **AT+BIND:<address>,<type>,<SrcAddress>,<SrcEP>, <ClusterID>,<DstAddress>[,<DstEP>]** Create Binding on a remote device with **<address>** the target Node's EUI64, NodeID, or Address Table entry **<type>** the Addressing mode shown as blow **<SrcAddress>** The EUI64 of the Source	Response **SEQ:XX** **OK** or **ERROR:<errorcode>** The response indicates success or failure in sending this message. The acknowledgement as well as the actual response to this request will follow as asynchronous prompts.

	Prompt **Bind:<NodeID>,<status>** In case of an error an status other than 00 will be displayed **<NodeID>** is the Remote node's NodeID. As with all unicasts after successful transmission the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not) the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed.
	Example **AT+BIND:0000,3,000d6f000059474e, 01,abcd,000D6F0000123456,01** **SEQ:01** **OK** **Bind:0000,00** **ACK:01** This command requests the target node to respond by listing its binding table starting from the requested index.

+UNBIND – Delete Binding on Remote Device	**(ZDO)**
Read Command **AT+UNBIND:<address>,<type>,<SrcAddress>,<SrcEP>,<ClusterID>,<DstAddress>[,<DstEP>]** Delete Binding on a remote device with **<address>** the target Node's EUI64, NodeID, or Address Table entry	Response **SEQ:XX** **OK or ERROR:<errorcode>** The response indicates success or failure in sending this message. The acknowledgement as well as the actual response to this request will follow as asynchronous prompts.

	Prompt **Unbind:<NodeID>,<status>** In case of an error an status other than 00 will be displayed **<NodeID>** is the Remote node's NodeID. As with all unicasts after successful transmission the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not) the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed.
	Example **AT+UNBIND:0000,3,000d6f000059474e,01,abcd, 000D6F0000123456,01** **SEQ:01** **OK** **Unbind:0000,00** **ACK:01** This command requests the target node to respond by listing its binding table starting from the requested index.

+PJOIN – Permit joining	
Read Command **AT+PJOIN:<sec>** **<sec> -** 16 bit hexadecimal number which represents the length of time in seconds during which the ZigBee coordinator or router will allow associations. The value 0x00 and 0xff indicate that permission is disabled or enabled, respectively, without a specified time limit. *Use on:* Coordinator or router. *Note:* SE network should not leave permit join on permanently, S0A can be set to switch off permit join.	Response OK or **ERROR<errorcode>** **<errorcode>** represents the error code (section 4).

+CBKE – Initiate CBKE With ESI Or A Partner Device	
Read Command **AT+CBKE:<Node ID>,<Endpoint>** **<Node ID> -** 16 bit hexadecimal number. The short address of the remote device which the local device (IPD) initiates CBKE with. e.g. 0000 for the Trust centre/ESI. **<Endpoint> -** 8 bit hexadecimal number. The Endpoint on the remote device. It should contain Key Establishment server cluster (This can be found by using AT+CLUSDISC command).	Response **OK** or **ERROR<errorcode>** **<errorcode>** represents the error code (section 4). **Prompt:** **PARTNERLINKREQ** (the prompts appear in establishing partner link key)

+RMKEY – Remove entry from the local link key table	
Execute Command **AT+RMKEY:<EUI>** **<EUI>** - 64 bit hexadecimal number, EUI of remote device which shall be removed from link key table.	Response **OK** or **ERROR:<errorcode>** **<errorcode>** represents the error code (section 4).

3.5.1.3 Cluster list

Cluster ID	Cluster
0000	Basic
000A	Time
0700	Price
0701	Demand Response and Load Control
0702	Simple Metering
0703	Messaging
0800	Key Establishment

3.5.1.4 ZigBee Discovery and Binding Commands

+DISCOVER – Discover SE Devices On The HAN	
Execute Command **AT+DISCOVER:<Cluster ID>** **<Cluster ID> -** 16 hexadecimal number which represents the cluster ID listed in section 3.5.1.3, The IPD can search for SE devices based on the specified cluster ID. *Notes:* Response shall show up all SE devices that match. Discovery will only search for server cluster.	Response **OK** or **ERROR:<errorcode>** (Please check section 4) Prompt**:** **<xxDEV>:<Node ID>,<End Point>** Carry out the ZigBee Service Discovery to find ZigBee SE devices that support the given match criteria. **<xxDEV> -** may be PRICEDEV, DRLCDEV, METERDEV or MSGDEV, they represent device that support price, DLRC, meter or message server cluster. **<Node ID> -** 16 bit hexadecimal number which represents network address of found device. **<End Point>** - 8 bit hexadecimal number which represents endpoint number on found remote device supporting the given match criteria.

+FINDMTR – Find All Meters On The HAN	
Execute Command **AT+FINDMTR** *Notes:* Response shall show up all SE metering devices.	Response **METERDEV:<Node ID>,<End Point>** **OK** or **ERROR:<errorcode>** (Please check section 4) This command carries out the ZigBee Service Discovery to find ZigBee SE metering devices. **<Node ID> -** 16 bit hexadecimal number which represents network address of found device**.** **<End Point>** - 8 bit hexadecimal number which represents endpoint number on found remote device supporting the given match criteria.

+CLUSDISC – Find All Supported Clusters On A Remote Device End Point	
Execute Command **AT+CLUSDISC:<Node ID>,<EndPoint>** **<Node ID>** - 16 bit hexadecimal number. The network address of the target device. **<EndPoint>** - 8 bit hexadecimal number.	Response **OK** **DISCCLUS:<Node ID>,<Status>,<EndPoint>** or **ERROR:<errorcode>** (see section 4) Prompt (will show, if <Status> is 00)

The end point of the target device. *Notes:* Response show up all supported clusters on specified device's endpoint.	**SERVER:<ClusterID>,<ClusterID>,<ClusterID>** **CLIENT:<ClusterID>,<ClusterID>,< ClusterID>**

+ATTRDISC - Find Supported Attributs On A Remote Device End Point	
Execute Command **AT+ATTRDISC:<NodeID>,<EP>,<ClusterID>,<AttributeID>,<MaxNumofAttr>** **<NodeID>** - 16 bit hexadecimal number represents target device's network address **<EP>** - 8 bit hexadecimal number represents target device's end point **<ClusterID>** - 16 bit hexadecimal number **<AttributeID>** -16 bit hexadecimal number the (discover will start from this attribute) **<MaxNumofAttr>** - 2 decimal number represent the number of attributes that required to be discovered, e.g: 01, 10	Response **OK** **DISCATTR:<Complete code>** **CLUS:<ClusterID>,ATTR:<AttributeID>,TYPE:<DataType>** or **ERROR:<errorcode>** (see section 4) Note: **<Complete code>** represents if all attributes support by this cluster have been discovered. 00 – Completed discovery 01 – Uncompleted discovery

+BINDMTR - Bind The IPD To A Meter/ESI	
Execute Command **AT+BINDMTR:<Node ID>,<End Point>** **<Node ID> - 16 bit hexadecimal number** **<End Point>** - 8 bit hexadecimal number *Use* Create a binding between the IPD and Meter/ESI. *Notes:* Valid end point is 00 to F0.	Response **OK** **BIND:<Node ID>,<Status>** or **ERROR:<errorcode>** <errorcode> represents the error code in section 4.

+UNBINDMTR - Unbind The IPD From The Meter/ESI	
Execute Command **AT+UNBINDMTR:<Node ID>,<End Point>** *Use* Break the binding between the IPD and Meter/ESI.	Response **OK** **UNBIND:<Node ID>,<Status>** or **ERROR:<errorcode>** <errorcode> represents the error code in section 4.

3.5.1.5 Time Control commands

+SETTIME - Set The Local Time	
Execute Command **AT+SETTIME:<year>,<month>,<day>,<hour>,<min>,<sec>** **AT+SETTIME: <time>** **<year>** - 4 digits decimal number **<month>** - 2 digits decimal number **<day>** - 2 digits decimal number **<hour>** - 2 digits decimal number **<min>** - 2 digits decimal number **<sec>** - 2 digits decimal number **<time> -** 32 bit hexadecimal number representing time in UTC format (number of seconds since 01.01.2000 00:00) *Examples:* **AT+SETTIME:2009,03,05,08,15,00** (2009,March 5th 8:15 AM)	Response **OK** or **ERROR:<errorcode>** **<errorcode>** please check section 4.

+GETTIME - Get The Local Time	
Execute Command **AT+GETTIME**	Response **TIME:<time>** **OK** or **ERROR:<errorcode>** (section 4) Returns current local time in UTC format. **<time> -** 32 bit hexadecimal value representing the local time, number of seconds since 0 hours, 0 minutes, 0 seconds, on the 1st of January, 2000 UTC.

+SYNCTIME - Synchronize The Local Time With The ESI (need binding table)	
Execute Command **AT+SYNCTIME:<Node ID>,<End Point>** **AT+SYNCTIME** **<Node ID>:**Target node address **<End Point>:**Target node's end point *Note:* The target shall support time sever cluster	Response **OK** **SYNCINGTIME:<time>** **or ERROR:<errorcode>** **<errorcode>** represents the error code in section 4.

3.5.1.6 ZCL Attributes commands

+READATR – Gets an Attribute From Specified Cluster Server	
Execute Command **AT+READATR:<NodeID>,<EP>,<Cluster>,<AttrID>** **AT+READATR:<Cluster>,<AttrID>** **<NodeID>** - 16 bit hexadecimal number, network address of a remote device. **<EP>** - 8 bit hexadecimal number, endpoint of a remote device. Valid end point is 0x01 to 0xF0. **<Cluster>** - 16 bit hexadecimal number which represents cluster ID, see section 3.5.1.3. **<AttrID>** - 16 bit hexadecimal number which represents attribute id according to ZigBee Smart Energy specification. *Note:* If destination address is not given then command is sent via binding table so that local binding shall be added first.	Response **OK** **RESPATTR:<Cluster>,<AttrID>,<Status>,<AttrInfo>** or **ERROR:<errorcode>** **<Cluster>** - cluster ID, 16 bit hexadecimal number, see section 3.5.1.3 **<AttrID>:** attribute id 16 bit hexadecimal number **<Status> -** 8 bit hexadecimal number which indicates the result of the requested operation. **<AttrInfo> -** hexadecimal number of char string (size depends on the attribute requested). <AttrInfo> shall only be valid if <Status> = 0x00. If <Status> indicates error, <AttrInfo> is not returned. **<errorcode>** represents the error code (section 4). Use this command to get the requested attribute from a cluster server on a remote SE device which supports this server cluster.

+READMATR – Gets Manufacturer Specific Attribute From Specified Cluster Server	
Execute Command **AT+READMATR:<NodeID>,<EP>,<Cluster>,<AttrID>** **AT+READMATR:<Cluster>,<AttrID>** **<NodeID>** - 16 bit hexadecimal number, network address of a remote device. **<EP>** - 8 bit hexadecimal number, endpoint of a remote device. Valid end point is 0x01 to 0xF0. **<Cluster>** - 16 bit hexadecimal number which represents cluster ID, see section 3.5.1.3. **<AttrID>** - 16 bit hexadecimal number which represents attribute id according to ZigBee Smart Energy	Response **OK** **RESPATTR:<Cluster>,<AttrID>,<Status>,<AttrInfo>** or **ERROR:<errorcode>** **<Cluster>** - cluster ID, 16 bit hexadecimal number, see section 3.5.1.3 **<AttrID>** - attribute id 16 bit hexadecimal number **<Status> -** 8 bit hexadecimal number which indicates the result of the

specification. *Note:* If destination address is not given then command is sent via binding table so that local binding shall be added first. Please set S60 before using this command.	requested operation. **<AttrInfo>** - hexadecimal number of char string (size depends on the attribute requested). <AttrInfo> shall only be valid if <Status> = 0x00. If <Status> indicates error, <AttrInfo> is not returned. **<errorcode>** represents the error code (section 4). Use this command to get the requested attribute from a cluster server on a remote SE device supporting this server cluster.

AT+WRITEATR – Sets An Attribute to Specified Cluster Client	
Execute Command	Response
AT+WRITEATR:<NodeID>,<EP>,<Cluster>,<AttrID>, <DataType>,<Data> **AT+WRITEATR:<Cluster>,<AttrID>,<Data Type>,<AttrValue>** **<NodeID>** - 16 bit hexadecimal number, network address of a remote device. **<EP>** - 8 bit hexadecimal number, endpoint of a remote device. Valid end point is 0x01 to 0xF0. **<Cluster> -** 16 bit hexadecimal number which represents cluster ID, see section 3.5.1.3 **<AttrID>** - 16 bit hexadecimal number which represents attribute id according to ZigBee Smart Energy specification. **<DataType> -** 8 bit hexadecimal number that represents the type of the data accepted by this Attribute (please check SE specification) **<AttrValue> -** If attribute value has an integer type this field shall contain hexadecimal representation in big-endian format. If attribute value has a string type this fiend contains sequence of characters. *Note:* If destination address is not given then command is sent via binding table so that local binding shall be added first.	**OK** **WRITEATTR:<Cluster>,<AttrID>,<Status>** or **ERROR:<errorcode>** **<Cluster>** - cluster ID, 16 bit hexadecimal number, see section 3.5.1.3 **<AttrID>:** attribute id 16 bit hexadecimal number **<Status>** - 8 bit hexadecimal number which indicates the result of the requested operation. If < Status > is not 00, it will be an errorcode which is listed in section 4. Use this command to set value to a writeable attribute on a remote SE device supporting this cluster.

AT+WRITEMATR – Sets Manufacturer Specific Attribute From Specified Cluster Server	
Execute Command	Response
AT+WRITEMATR:<NodeID>,<EP>,<Cluster>,<AttrID>,<DataType>,<Data> **AT+WRITEMATR:<Cluster>,<AttrID>,<Data Type>,<AttrValue>** **<NodeID>** - 16 bit hexadecimal number, network address of a remote device. **<EP> -** 8 bit hexadecimal number, endpoint of a remote device. Valid end point is 0x01 to 0xF0. **<Cluster>** - 16 bit hexadecimal number which represents cluster ID, see section 3.5.1.3. **<AttrID>** - 16 bit hexadecimal number which represents attribute id according to ZigBee Smart Energy specification. **<DataType>** - 8 bit hexadecimal number that represents the type of the data accepted by this Attribute (please check SE specification) **<AttrValue>** - If attribute value has an integer type this field shall contain hexadecimal representation in big-endian format. If attribute value has a string type this fiend contains sequence of characters. *Note:* If destination address is not given then command is sent via binding table so that local binding shall be added first. Please set S60 before using this command.	**OK** **WRITEATTR:<Cluster>,<AttrID>,<Status>** or **ERROR:<errorcode>** **<Cluster>** - cluster ID, 16 bit hexadecimal number, see section 3.5.1.30 **<AttrID>** - attribute id 16 bit hexadecimal number **<Status>** - 8 bit hexadecimal number which indicates the result of the requested operation. If < Status > is not 00, it will be an errorcode which is listed in section 4. Use this command to write attribute values to a writeable attribute on a remote SE device supporting this cluster.

+SETATR – Set Value Of A Local ZCL Attribute	
Execute Command	Response
AT+SETATR:<ClusterID>,<AttrID>,<AttrValue> **<ClusterID>** - 16 bit hexadecimal number which represents cluster ID, see section 3.5.1.3 **<AttrID>** - 16 bit hexadecimal number which represents attribute id (Please refer to ZigBee Home Automation specification). **<AttrValue>** - If attribute value has an integer type this field shall be in big-endian format. If attribute value has a string type, this fiend contains sequence of characters. Maximum length is 110 characters.	**OK** or **ERROR:<errorcode>** **<errorcode>** represents the error code (section 4). Use this command to set a new value to attribute from a cluster on a local device supporting this cluster.

+GETATR – Get value of a local ZCL attribute	
Execute Command	Response
AT+GETATR:<Cluster>,<AttrID> **<Cluster>** - 16 bit hexadecimal number which represents cluster ID, see section 3.5.1.3 **<AttrID>** - 16 bit hexadecimal number which represents attribute ID according to ZigBee Home Automation specification.	**ATTR:<AttrValue>** **OK** or **ERROR:<errorcode>** **<AttrValue>** - If attribute value has an integer type this field shall contain hexadecimal representation. If attribute value has a string type this fiend contains sequence of characters. Maximum length is 110 characters. **<errorcode>** represents the error code (section 4). Use this command to get a value of an attribute from a cluster server on a local device supporting this cluster.

3.5.2 IPD specified command set

3.5.2.1 Simple Metering cluster - Client

+GMP – Send Get Meter Profile request	
Execute Command	Response
AT+GMP:<NodeID>,<EP>,<intervalchannel>,<end time>,<number of periods> **AT+GMP:<intervalchannel>,<endtime>,<number of periods>** **<NodeID>** - 16 bit hexadecimal number, network address of a remote device. **<EP>** - 8 bit hexadecimal number, endpoint of a remote device. Valid end point is 0x01 to 0xF0. **<intervalchannel>** - 1 decimal digit, which is an enumerated value used to select the quantity of interest returned by the Get Profile Response Command. The Interval Channel values are listed in Table: 0 – Consumption Delivered 1 – Consumption Received **<end time> -** 32 bit hexadecimal number (in UTCTime) used to select an Intervals block from	**OK** or **ERROR:<errorcode>** **<errorcode>** check section 4. **Profile:<NodeID>,<EP>,<EndTime>,<Status>,<ProfileIntervalPeriod>,<NumberOfPeriod>,<Interval1>,<Interval2>,…,<IntervalN>** **<NodeID>** - 16 bit hexadecimal number, network address of a remote device. **<EP>** - 8 bit hexadecimal number, endpoint of a remote device. Valid end point is 0x01 to 0xF0. **<EndTime>** - 32 bit value (in UTC) representing the end time of the most chronologically recent interval being requested. Example: Data collected from 2:00 PM to 3:00 PM would be specified as a 3:00 PM interval (end time). It is important to note that the current interval accumulating is not included in most recent block but can be retrieved using

all the Intervals blocks available. The Intervals block returned is the most recent block with its EndTime equal or older to the one provided. The most recent Intervals block is requested using an End Time set to 0x00000000, subsequent Intervals block are requested using an End time set to the EndTime of the previous block - (number of intervals of the previous block*ProfileIntervalPeriod).

<number of periods> - 8 bit hexadecimal number represents the number of intervals being requested. This value can't exceed the size stipulated in the MaxNumberOfPeriodsDelivered attribute. If more intervals are requested than can be delivered, the GetProfileResponse will return the number of intervals equal to MaxNumberOfPeriodsDelivered. If fewer intervals available for the time period, only those available are returned.

Note:

If NodeId and EP are not present, the device uses binding to send this command.

the CurrentPartialProfileIntervalValue attribute.

<Status> - 8 bit hexadecimal value. Table below lists the valid values returned in the Status field.

0x00 - Success
0x01 - Undefined Interval Channel requested
0x02 - Interval Channel not supported
0x03 - Invalid End Time
0x04 - More periods requested than can be returned
0x05 - No intervals available for the requested time
0x06 to 0xFF Reserved for future use.

<ProfileIntervalPeriod>: 8 bit hexadecimal value representing the interval or time frame used to capture metered Energy, Gas, and Water consumption for profiling purposes. ProfileIntervalPeriod is an enumerated field representing the following timeframes:

0x00 - Daily
0x01 - 60 minutes
0x02 - 30 minutes
0x03 - 15 minutes
0x04 - 10 minutes
0x05 - 7.5 minutes
0x06 - 5 minutes
0x07 - 2.5 minutes

<NumberofPeriodsDelivered> - 8 bit hexadecimal value representing the number of intervals the device is returning. Please note the number of periods returned in the Get Profile Response command can be calculated when the packets are received and can replace the usage of this field. The intent is to provide this information as a convenience.

<Intervals>: Series of 24 bit hexadecimal values representing interval data captured using the period specified by the ProfileIntervalPeriod field. The content of the interval data depend of the type of information requested using the Channel field in the Get Profile Command. Data is organized in a reverse chronological order, the most recent interval is transmitted first and the oldest interval is transmitted last. Invalid intervals should be marked as 0xFFFFFF.

See ZigBee Smart Energy specification [6] for fields' description.

3.5.2.2 Price Cluster - Client

+PRICELBL - Get The Label Assigned To The Price Tier From The ESI	
Execute Command **AT+PRICELBL:<NodeID>,<EP>,<Tier N>** **AT+PRICELBL:<Tier N>** **<Tier N>** - decimal number 1 to 6 *Use:* Get the label assigned to the price tier declared in the price received from the ESI (if binding has been established). Or specify node ID and end point to get price label. *Notes:* **<PriceLabel>** is returned only if <Status>=0x00	Response **OK** **PRICELABEL:<AttributeID>,<Status>[,PriceLabel]** or **ERROR:<errorcode>** **<AttributeID>** - 16-bit hexadecimal number represents the attribute id of this Tier **<Status>** - 8-bit hexadecimal number, if it is not 00, it will be a error code(please check section 4) **<Price Label>** - Character string **<errorcode>** - the code explained in section 4.

+CURPRICE - Get The Current Pricing From The ESI	
Execute Command **AT+CURPRICE:<NodeID>,<EP>,<Options>** **AT+CURPRICE: <Options>** **<NodeID>** - 16 bit hexadecimal number, network address of a remote device. **<EP>** - 8 bit hexadecimal number, endpoint of a remote device. Valid end point is 0x01 to 0xF0. **<Options>** - one decimal digit which if is 0 – don't receive price broadcasts from the ESI, if is 1 – receive price broadcasts from the ESI. *Note:* If destination address is not given then command is sent via binding table so that local binding shall be added first.	Response **OK** **PRICE:<NodeID>,<EP>,<ProviderID>,<RateLabel>,<EventID>,<CurrentTime>,<UnitofMeasure>,<Currency>,<PriceTrailingDigit&PriceTier>,<NoofPriceTiers&RegisterTier>,<StartTime>,<DurationinMinutes>,<Price>,<PriceRatio>,<GenerationPrice>,<GenerationPriceRatio>,<AlternateCostDelivered>,<AlternateCostUnit>,<AlternateCostTrailingDigit>,<NumberOfBlockThresholds>,<PriceControl>** or **ERROR:<errorcode>** (please see section 4) Get the pricing information from the ESI for the current time. See ZigBee Smart Energy specification [6] for fields' description. **<NodeID>** - 16 bit hexadecimal number, network address of a remote device. **<EP>** - 8 bit hexadecimal number, endpoint of a remote device. Valid end point is 0x01 to 0xF0. **<Provider ID>** - 32 bit hexadecimal number

	<Rate Label> - Character string **<Event ID>**- 32 bit hexadecimal number **<Current Time>**- 32 bit hexadecimal number **<Unit of Measure>**- 8 bit enumeration **<Currency>**- 16 bit hexadecimal number **<PriceTrailingDigit&PriceTier>**- 8 bit bitmap **<NoofPriceTiers&RegisterTier>**- 8 bit bitmap **<StartTime>**- 32 bit hexadecimal number **<DurationinMinutes>**- 16 bit hexadecimal number **<Price>**- 32 bit hexadecimal number **<Price Ratio>**- 8 bit hexadecimal number **<Generation Price>**- 32 bit hexadecimal number **<GenerationPriceRatio>**- 8 bit hexadecimal number **<AlternateCostDelivered>**- 32 bit hexadecimal number **<AlternateCostUnit>**- 8 bit hexadecimal number **<AlternateCostTrailingDigit>**- 8 bit hexadecimal **<NumberOfBlockThresholds>**- 8 bit hexadecimal **<Price Control>** - 8 bit hexadecimal number

+SCHPRICE - Get All The Scheduled Pricing From The ESI	
Execute Command	Response
AT+SCHPRICE:<NodeID>,<EP>,<StartTime>,<Num> **AT+SCHPRICE: <StartTime>,<Num>** **<NodeID>** - 16 bit hexadecimal number, network address of a remote device. **<EP>** - 8 bit hexadecimal number, endpoint of a remote device. Valid end point is 0x01 to 0xF0. **<StartTime>** - 8 bit hexadecimal number representing number of minutes +/- now. e.g.: +02 means 2 minutes later from now and -10	**OK** **PRICE:<NodeID>,<EP>,<ProviderID>,<RateLabel>,<EventID>,<CurrentTime>,<UnitofMeasure>,<Currency>,<PriceTrailingDigit&PriceTier>,<NoofPriceTiers&RegisterTier>,<StartTime>,<DurationinMinutes>,<Price>,<PriceRatio>,<GenerationPrice>,<GenerationPriceRatio>,<AlternateCostDelivered>,<AlternateCostUnit>,<AlternateCostTrailingDigit>,<NumberOfBlockThresholds>,<PriceControl>** or **ERROR:<errorcode>** Get all currently scheduled pricing information

means 16 minutes before **<Num>** - 8 bit hexadecimal number, number of returning prices, 00 means all available prices. *Examples:* **AT+SCHPRICE:+02,01** **AT+SCHPRICE:0000,aa,-10,00** *Note:* **Price:** prompt will show up for all the prices received from the ESI. If destination address is not given then command is sent via binding table so that local binding shall be added first.	from the ESI. See ZigBee Smart Energy specification for fields description. **<NodeID>**- 16 bit hexadecimal number, network address of a remote device. **<EP>**- 8 bit hexadecimal number, endpoint of a remote device. Valid end point is 0x01 to 0xF0. **<ProviderID>**- 32 bit hexadecimal number **<RateLabel>**- Character string **<EventID>**- 32 bit hexadecimal number **<CurrentTime>**- 32 bit hexadecimal number **<UnitofMeasure>**- 8 bit enumeration **<Currency>**- 16 bit hexadecimal number **<PriceTrailingDigit&PriceTier>**- 8 bit bitmap **<NoofPriceTiers&RegisterTier>**- 8 bit bitmap **<StartTime>**- 32 bit hexadecimal number **<DurationinMinutes>**- 16 bit hexadecimal number **<Price>**- 32 bit hexadecimal number **<PriceRatio>**- 8 bit hexadecimal number **<GenerationPrice>**- 32 bit hexadecimal number **<GenerationPriceRatio>**- 8 bit hexadecimal number **<AlternateCostDelivered>**- 32 bit hexadecimal number **<AlternateCostUnit>**- 8 bit hexadecimal number **<AlternateCostTrailingDigit>**- 8 bit hexadecimal **<NumberOfBlockThresholds>**- 8 bit hexadecimal **<PriceControl>** - 8 bit hexadecimal number **<errorcode>** represents the error code.

3.5.2.3 Messaging Cluster - Client

+LASTMSG - Get The Last Message From The ESI	
Execute Command **AT+LASTMSG:<NodeID>,<EP>** **AT+LASTMSG** **<NodeID>** - 16 bit hexadecimal number, network address of a remote device. **<EP>** - 8 bit hexadecimal number, endpoint of a remote device. Valid end point is 0x01 to 0xF0. *Note:* If the received message requires confirmation use AT+LASTMSG. The current active message should be replaced if a new message is received from the ESI. If destination address is not given then command is sent via binding table so that local binding shall be added first.	Response **OK** **Message:<MessageID>,<MessageControl>,<StartTime>,<DurationinMinutes>,<Message>** or **ERROR:<errorcode>** **<Message ID>** - 32 bit hexadecimal number **<Message Control>** - 8 bit bitmap **<Start Time>** - 32 bit hexadecimal number **<Duration in Minutes>** - 16 bit hexadecimal number **<Message>** - Character string **<errorcode>** represents the error code (section 4). See ZigBee Smart Energy specification [6] for fields' description.

+ACKMSG - Acknowledge A Message	
Execute Command **AT+ACKMSG:<NodeID>,<EP>,<MessageID>** **AT+ACKMSG:<Message ID >** **<NodeID>** - 16 bit hexadecimal number, network address of a remote device. **<EP>** - 8 bit hexadecimal number, endpoint of a remote device. Valid end point is 0x01 to 0xF0. **<Message ID>** - 32 bit hexadecimal number, message unique identifier. *Note:* If destination address is not given, command is sent via binding table so that local binding shall be added first.	Response **OK** or **ERROR:<errorcode>** Send an acknowledgement for the last message received from the ESI. **<errorcode>** represents the error code (please check section 4).

Message control field bit map			
Bits	**Enumeration**	**Value**	**Description**
Bits 0 to 1	Normal transmission only	0	Send message through normal command
	Normal and Anonymous Inter-PAN transmission	1	Send message through normal command function to client and pass message on to the Anonymous Inter-PAN transmission mechanism.
	Anonymous Inter-PAN transmission only	2	Send message through the Anonymous Inter-PAN transmission mechanism.
	Reserved	3	Reserved value for future use.
Bits 2 to 3	Low	0	Message to be transferred with a low level of importance.
	Medium	1	Message to be transferred with a medium level of importance.
	High	2	Message to be transferred with a high level of importance.
	Critical	3	Message to be transferred with a critical level of importance.
Bits 4 to 6	Reserved	N/A	These bits are reserved for future use.
Bit 7	Message Confirmation	0	Message Confirmation not required.

Table 7 Message control field bit map

3.5.2.4 Demand Response and Load Control Cluster - Client

+RES – Send DRLC cluster command "Report Event Status"

Execute Command	Response
AT+RSE:<NodeID>,<EP>,<eventid>,<status>,<criticallevel>,<dutycycle>,<control byte> **AT+RSE:<eventid>,<status>,<criticallevel>,<dutycycle>,<controlbyte>** **<NodeID>** - 16 bit hexadecimal number, network address of a remote device. **<EP>** - 8 bit hexadecimal number, endpoint of a remote device. Valid end point is 0x01 to 0xF0. **<event id>** - 16 bit hexadecimal number represents unique identifier which allows matching of Event reports with a specific Demand Response and Load Control event. **<criticality>** - 8 bit hexadecimal number represents criticality level value applied by the device, see the corresponding field in the Load Control Event Command for more information. **<duty>** - 8 bit hexadecimal number defines the maximum On state duty cycle as a percentage of time. Example, if the value is 80, the device	**OK** or **ERROR:<errorcode>**

would be in an "on state" for 80% of the time for the duration of the event. Range of the value is 0 to 100. A value of 0xFF indicates the field is not used. All other values are reserved for future use. **<controlbyte>** - 8 bit hexadecimal number identifies additional control options for the event. The Bit Map for this field: Bit 0 – 1:Randomize Start time, 0:Randomized Start not Applied Bit 1 – 1: Randomize End time, 0:Randomized End not Applied Bits 2 to 7 – Reserved *Note:* If node ID and endpoint are not specified, command will be sent via binding table. So local binding should be created before using binding table for sending this command.	

+OPT – Opts In To The DRLC Event Or Opts Out of DRLC Indicated By Event ID.	
Execute Command **AT+OPT:<in or out>,<event id>** **<in or out>** - 0 represents out, 1 represents in. **<event id>** - 8 bit hexadecimal number	Response **OK** or **ERROR:<errorcode>**

+GSE – Get Scheduled Events From ESI	
Execute Command **AT+GSE:<NodeID>,<EP>,<starttime>,<numberofevents>** **AT+GSE:<starttime>,<numberofevents>** **<NodeID>** - 16 bit hexadecimal number, network address of a remote device. **<EP>** - 8 bit hexadecimal number, endpoint of a remote device. Valid end point is 0x01 to 0xF0. **<starttime>** - 32 bit hexadecimal UTC Timestamp representing the starting time for any scheduled events to be re-sent. **<number of events>** - 8 bit hexadecimal number represents the maximum number of	Response **OK** **DRLCRECEIVED:<eventid>,<starttime>,<duration>,<deviceclass>,<group>,<criticality>,<coolingTempOffset>,<heatingTempOffset>,<coolingTempSetPoint>,<heatingTempSetPoint>,<avgLoadAdjustmentPercent>,<duty>,<control byte>** or **ERROR:<errorcode>** **<errorcode>** represents the error code (please check section 4).

events to be sent. A value of 00 would indicate all available events are to be returned.	It may also show following prompt: **DRLCSTART:** **DRLCCOMPLETE:** **DRLCOPTOUT:** **DRLCOPTIN:** **DRLCCANCELED:** **DRLCSUPERSEDED:** Followed by: **<eventid>,<starttime>,<duration>,<deviceclass>,<group>,<criticality>,<coolingTempOffset>,<heatingTempOffset>,<coolingTempSetPoint>,<heatingTempSetPoint>,<avgLoadAdjustmentPercent>,<duty>,<controlbyte>** or **DRLCEVENT:<eventstatus>,<eventid>,<starttime>,<duration>,<deviceclass>,<group>,<criticality>,<coolingTempOffset>,<heatingTempOffset>,<coolingTempSetPoint>,<heatingTempSetPoint>,<avgLoadAdjustmentPercent>,<duty>,<control byte>**

3.5.2.4.1 Prompts

Load Control Event	
<eventstatus> - 8 bit hexadecimal number. Note: <event status> will be shown with the prompt **DRLCEVENT.** 08 - Event partially completed with User "Opt-Out" 09 - Event partially completed due to User "Opt-In" 0A- Event completed, no User participation (Previous "Opt-Out") F8 - Rejected - Invalid Cancel Command (Default) F9 - Rejected - Invalid Cancel Command (Invalid Effective Time) FB- Rejected - Event was received after it had expired FD- Rejected - Invalid Cancel Command (Undefined Event) FE- Load Control Event command Rejected <**event id>** - 16 bit hexadecimal number represents unique identifier which allows matching of Event reports with a specific Demand Response and Load Control event. **<start time>** - 8 bit hexadecimal number represents shift in minutes from now when the event is scheduled to start. A start time of 0x00000000 is a special time denoting "now".	Appears when a new load control event is received. *Note:* To receive load control events remote binding shall be added to the ESI (see AT+BIND command).

<duration> - 8 bit hexadecimal number represents duration of this event in number of minutes. Maximum value is 0x5A0 (one day).

<device class> - 8 bit encoded field in hexadecimal format representing the Device Class to apply the current Load Control Event. Each bit, if set individually or in combination, indicates the class device(s) needing to participate in the event.

Device Class Field BitMap/Encoding:
Bit 0 - HVAC compressor or furnace
Bit 1 - Strip Heaters/Baseboard Heaters
Bit 2 - Water Heater
Bit 3 - Pool Pump/Spa/Jacuzzi
Bit 4 - Smart Appliances
Bit 5 - Irrigation Pump
Bit 6 - Managed Commercial & Industrial (C&I) loads
Bit 7 - Simple misc. (Residential On/Off) loads
Bit 8 - Exterior Lighting
Bit 9 - Interior Lighting
Bit 10 - Electric Vehicle
Bit 11 - Generation Systems
Bit 12 to 15 Reserved

<group> - 8 bit hexadecimal number represents utility enrolment group which can be used in conjunction with the Device Class bits. It provides a mechanism to direct Load Control Events to groups of Devices.

<criticality> - 8 bit hexadecimal number defines the level of criticality of this event.

Criticality Levels

0 – Reserved
1 – Green
2 – 1
3 – 2
4 – 3
5 – 4
6 – 5
7 – Emergency
8 – Planned Outage
9 – Service Disconnect
0x0A to 0x0F – Utility Defined
0x10 to 0xFF – Reserved

<coolingTempOffset> - 8 bit hexadecimal value representing requested offset to apply to the normal cooling setpoint at the time of the start of the event in + 0.1 ºC. For more details see [6].

<heatingTempOffset> - 8 bit hexadecimal value representing requested offset to apply to the normal heating setpoint at the time of the start of the event in + 0.1 ºC. For more details see [6].

<coolingTempSetPoint> -16 bit hexadecimal value representing requested cooling set point in 0.01 degrees Celsius. The value 0x8000 means that this field has not been used by the end device. For more details see [6].

<heatingTempSetPoint> - 16 bit hexadecimal value representing requested heating set point in 0.01 degrees Celsius. The value 0x8000 means that this field has not been used by the end device. For more details see [6].

<avgLoadAdjustmentPercent> - 16 bit hexadecimal value representing a maximum energy usage limit as a percentage of the client implementations specific average energy usage. The load adjustment percentage is added to 100% creating a percentage limit applied to the client implementations specific average energy usage. A -10% load adjustment percentage will establish an energy usage limit equal to 90% of the client implementations specific average energy usage. Each load adjustment percentage is referenced to the client implementations specific average energy usage. There are no cumulative effects. The range of this field is -100 to +100 with a resolution of 1 percent. A -100% value equals a total load shed. A +100% value will limit the energy usage to the client implementations specific average energy usage.

A value of 0x80 indicates the field is not used. All other values are reserved for future use.

<duty> - 8 bit hexadecimal number defines the maximum On state duty cycle as a percentage

of time. Example, if the value is 80, the device would be in an “on state” for 80% of the time for the duration of the event. Range of the value is 0 to 100. A value of 0xFF indicates the field is not used. All other values are reserved for future use.

<control byte> - 8 bit hexadecimal number identifies additional control options for the event.

The BitMap for this field:

0 – 1 = Randomize Start time, 0 = Randomized Start not Applied

1 – 1 = Randomize End time, 0 = Randomized End not Applied

2 to 7 – Reserved

4 List of Status codes

00	Everything OK - Success
01	Fatal Error
02	Unknown command
04	Invalid S-Register
05	Invalid parameter
06	Recipient could not be reached
07	Message was not acknowledged
0C	Too many characters
0D	License Problem
0E	PAN could not be established because duplicate PAN ID was detected
0F	Fatal error initialising the network
10	Error bootloading
12	Fatal error initialising the stack
18	Node has run out of Buffers
19	Trying to write read-only register
20	Invalid password
25	Cannot form network
27	No network found
28	Operation cannot be completed if node is part of a PAN
2C	Error leaving the PAN
2D	Error scanning for PANs
33	No response from the remote boot loader
39	MAC transmit queue is full
6C	Invalid binding index
66	Message is not been sent successfully
70	Invalid operation
74	Message too long
86	Unsupported attribute
87	Invalid value
88	Read only
8A	Duplicate Exist
8B	Not found attribute
8D	Incorrect attribute type
91	Operation only possible if joined to a PAN
93	Node is not part of a Network
94	Cannot join network
96	Mobile End Device Move to new Parent Failed
98	Cannot join ZigBee 2006 Network as Router
A3	Invalid Endpoint
A6	Error in trying to encrypt at APS level. No link key entry in the table for the destination
AB	Trying to join, but no beacons could be heard
AC	Network key was sent in the clear when trying to join secured
AD	Did not receive Network Key
AE	No Link Key received
AF	Preconfigured Key Required
B1	Not attached to a meter
B2	ESI end point not known
C5	NWK Already Present
C7	NWK Table Full
C8	NWK Unknown Device

5 S-Registers

Most S-Registers of the modules can be read and written. All s-registers are stored in non-volatile memory and will keep their user defined settings unless reset to the factory defaults using the “AT&F” command. The S-Registers used in this firmware are summarized in the table below.

S-Register Overview		Local R/W
S00	Channel Mask	(•/•)
S01	Transmit Power Level	(•/•)
S02	Preferred PAN ID	(•/•)
S03	Preferred Extended PAN ID	(•/•)
S04	Local EUI	(•/-)
S05	Local NodeID	(•/-)
S06	Parent’s EUI	(•/-)
S07	Parent’s NodeID	(•/-)
S08	Network Key[1]	(-/•)
S09	Link Key[1]	(-/•)
S0A	Main Function[1]	(•/•)
S0B	User Readable Name[1]	(•/•)
S0C	Password[1]	(-/•)
S0D	Device Information	(•/-)
S12	UART Setup	(•/•)
S60	Manufacture Code	(•/•)
S61	Installation Code	(•/•)

Table 8: S-Register Overview (Note: “1” means password Protected Registers)

5.1 S-Registers for Network Setup

S00 – Channel Mask

Description

The 802.15.4 channel mask.

Operations

R/W LOCAL

Becomes effective

when Joining, Scanning or establishing a PAN

Note

The channel mask does not affect the AT+JPAN command

Storage

Non-Volatile

Parameters

XXXX

Where XXXX represents a 16-bit decimal number enabling IEEE 802.15.4 channel numbers 11 to 26. Writing a bit to 1 enables a channel and subsequently writing a bit to 0 disables a channel for scanning, joining and establishing networks. e.g. when setting S00 to 0001, only channel 11 will be used for all following operations.

Range

0001 - FFFF

Factory Default

FFFF or 7FFF(for LRS)

S01 – Transmit Power Level

Description

The device’s transmit power level in dBm.

Operations

R/W LOCAL

Notes

The output power of the”-PA” and “-LR” variants is higher than the value in S01. Please refer to the respective hardware manuals.

Becomes effective

When Joining or establishing a PAN

Storage

Non-Volatile

Parameters

snn

Where snn represents a signed 8-bit decimal number.

Range

ETRX3: 8 to -43

ETRX3 LRS Variants: -7 to -43

Actual values are {8, 7, 6, 5, 4, 3, 2, 1, -1, -2, -3, -4, -5, -6, -7, -8, -9, -11, -12, -14, -17, -20, -26, -43} Entering a value not on this list (such as –19) will result in the next lowest output power.

Entering a value higher than 3 will automatically enable boost mode regardless of the setting of bit E of S11.

Factory Default

ETRX3 LRS-Variants: -11

Others: 3

Factory Default

3 or (-11 for LRS)

S02 – Preferred PAN ID

Description

The 802.15.4 PAN ID.

Operations

R/W LOCAL

Becomes effective

When Joining or establishing a PAN

Notes

Two networks operating on the same channel with the same PANID, but a different EPID are detected to be in conflict with each other. PANID conflicts are detected by the stack and resolved by one of the networks dynamically changing its PAN ID.

The preferred PID does not affect the AT+JPAN command

Storage

Non-Volatile

Parameters

<PID>

Where <PID> represents a 16-bit hexadecimal number

Range

0000 – FFFF

When establishing a PAN the coordinator will pick a random PAN ID if S02 is set to 0000. If set to any value between 0001 and FFFF this number will be used as PAN ID instead, unless trying to use a PAN ID which already exists on the same channel. In this case a random PAN ID will be used instead.

When joining only a PAN with the ID stored in S02 will be joined unless S02 is set to 0000. In this case the next best PAN which allows joining is joined.

Factory Default

0000

S03 – Preferred Extended PAN ID

Description The extended PAN ID. Operations **R/W LOCAL** Becomes effective **When Joining or establishing a PAN** *Note* The EPID is used for PAN ID conflict detection. It is therefore recommended to use a random EPID at all times. The preferred EPID does not affect the AT+JPAN command Storage **Non-Volatile**	Parameters **<EPID>** Where <EPID> represents a 64-bit hexadecimal number Range **0000000000000000 – FFFFFFFFFFFFFFFF** When establishing a PAN the coordinator will pick a random EPID if S03 is set to all 0's. If set to any other value this number will be used as EPID instead. When joining only a PAN with the EPID stored in S03 will be joined unless S03 is set to all 0's. In this case the next best PAN which allows joining is joined. Factory Default **0000000000000000**

S04 – Local EUI64

Description The local node's unique EUI64 identifier. Operations **R LOCAL** Storage **Non-Volatile**	Parameters **<EUI64>** Range **0000000000000000 – FFFFFFFFFFFFFFFF** Factory Default **<unique number>**

S05 – Local 16-Bit NodeID

Description The local node's 16-bit NodeID. *Note* Reading this register while not associated with a network will result in an undefined return value. Operations **R LOCAL** Storage **Non-Volatile**	Parameters **<NodeID>** Range **0000-FFFF** Factory Default **n/a**

S06 – Parent's EUI64

Description The parent node's unique EUI64 identifier. *Note* The return value is undefined for nodes	Parameters **<EUI64>** Range

without parents (coordinator and nodes that are not joined to a network) Operations **R LOCAL** Storage **Non-Volatile**	**0000000000000000 – FFFFFFFFFFFFFFFF** Factory Default **n/a**

S07 – Parent's 16-Bit NodeID	
Description The parent node's 16-bit NodeID. Operations **R LOCAL** *Note* The return value is undefined for nodes without parents (coordinator and nodes that are not joined to a network) Storage **Non-Volatile**	Parameters **<NodeID>** Range **0000-FFFF** Factory Default **n/a**

S08 – Network Key	
Description The network key which can be written using the password. The default password for R3xx is "password". Operations **W LOCAL** Write operation **ATS08=<key>:<password>** Becomes effective **Only when establishing a PAN** Storage **Non-Volatile**	Range **From 0 to 2^{128}-1** The 128-bit AES network key in hexadecimal representation (32 characters). When set to all 0's (default) a random network key is generated when establishing a PAN. This key is transmitted to all joining nodes and can be encrypted using the link key. Factory Default **aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa**

S09 – Trust Centre Link Key	
Description The link key which can be written using the password. The default password for R1xx is "password". Operations **W LOCAL** Write operation	Range **From 0 to 2128-1** The 128-bit trust centre link key in hexadecimal representation (32 characters). When set to all 0s (default) a random trust centre link key is generated when establishing a PAN.

ATS09=<key>:<password> Becomes effective **When Joining or establishing a PAN** Storage **Non-Volatile**	Factor Default **00000000000000000000000000000000**

5.2 S-Registers for Module Setup

S0A – Main Function	
Description: Defines the behavior of the Device. *Operations:* **R/W LOCAL** *Write operation:* **ATS0A=XXXX:<Password>** *Note:* For security reasons this register is password protected. The default password is "password". *Storage:* **Non-Volatile**	Parameters **XXXX** Where XXXX represents a 16-bit hexadecimal number. Range **0000 to FFFF** **Bit 0:** Set to control permit joining flag. Set to 0 to allow other nodes to join the network using this node as their parent **Bit 1-6:** Reserved **Bit 7:** Device uses preconfigured link key when joining **Bit 8:** Device uses install code stored in S61 to generate preconfigured link key when joining (for joining device only) **Bit 9-F:** Reserved If both Bit 7 and 8 are set to 0. Joining device will use install code stored in manufacture token for joining. Factory Default **0001**

S0B – User Readable Name	
Description Password protected user defined name which can be used to identify the node Operations **R/W LOCAL** Write operation **ATS0B=<name>:<password>** Becomes effective **Instantly.** Storage **Non-Volatile**	Parameters **0123456789ABCDEF** Name with up to 16 characters.

S0C – Password	
Description The local node's password. Operations **W LOCAL** Write operation **ATS0C=<NEW>:<OLD>** Becomes effective **Instantly.** Storage **Non-Volatile**	Parameters **12345678** 8 case sensitive characters (8 bytes). **Note that the password must have exactly 8 characters.** Factory Default **password**

S0D – Firmware Revision	
Description String containing the module's order code and firmware revision. Operations **R LOCAL** Storage **Non-Volatile**	Parameters ccc…ccc Text string Example ETRX357R300X Factory Default **N/A**

S0F – Prompt Enable	
Description Enable/Disable default response prompts. Operations **R/W LOCAL** Becomes effective **Instantly** Storage **Non-Volatile**	Parameters **XXXX** XXXX represents a 16-bit hexadecimal number. **Bit A:** Set: Show received default response **All the other bits are reserved** Factory Default **0400**

S60 – Manufacture Code	
Description String containing the manufacture code input by user. This code is only used for reading or writing manufacture defined attribute on a remote node. *Operations* **R/W LOCAL** *Storage* **Non-Volatile**	Parameters **XXXX** XXXX represents a 16-bit hexadecimal number Factory Default **N/A**

S61 – Installation Code	
Description String containing the installation code input by user. This code can be used to generate preconfigured link key for a joining device to join a network. The code should include a CRC code Please check SE specification 1.1 for more information	Parameters ccc…ccc Text string
Operations **R/W LOCAL**	
Storage **Non-Volatile**	Factory Default **N/A**

5.3 I/O related S-Registers

S12 – UART Setup	
Description The device's RS232 Baudrate and mode. The default setting of 0500 results in: 19200bps, no parity, 1 stop bit, 8 data bits.	Parameters **XXXX** Where XXXX represents a 16-bit hexadecimal number.
Operations **R/W LOCAL** Becomes effective Instantly. *Note* If bit 5 is set, bi-directional Hardware Flow Control is used instead of XON/XOFF flow control. Note that in case the 128-byte output buffer of the module is full data will be dropped.	Range of the most significant byte **00 to 0C** 00: 1200 baud 01: 2400 baud 02: 4800 baud 03: 9600 baud 04: 14400 baud 05: 19200 baud 06: 28800 baud 07: 38400 baud 08: 50000 baud 09: 57600 baud 0A: 76800 baud 0B: 100000 baud 0C: 115200 baud
Storage **Non-Volatile**	Factory Default **0500**

6 Interpreting RSSI Energy Levels

The readings from “AT+ESCAN” represent the hexadecimal numbers with offset by +127 to make them positive numbers.

The RSSI is calculated over an 8-symbol period as well as at the end of a received packet. Algorithm utilizes the RX gain settings and the output level of the ADC. The linear range of RSSI is specified to be 40dB over all temperatures. At room temperature, the linear range is approximately 60dB (-90 dBm to -30dBm).

7 Firmware upgrades

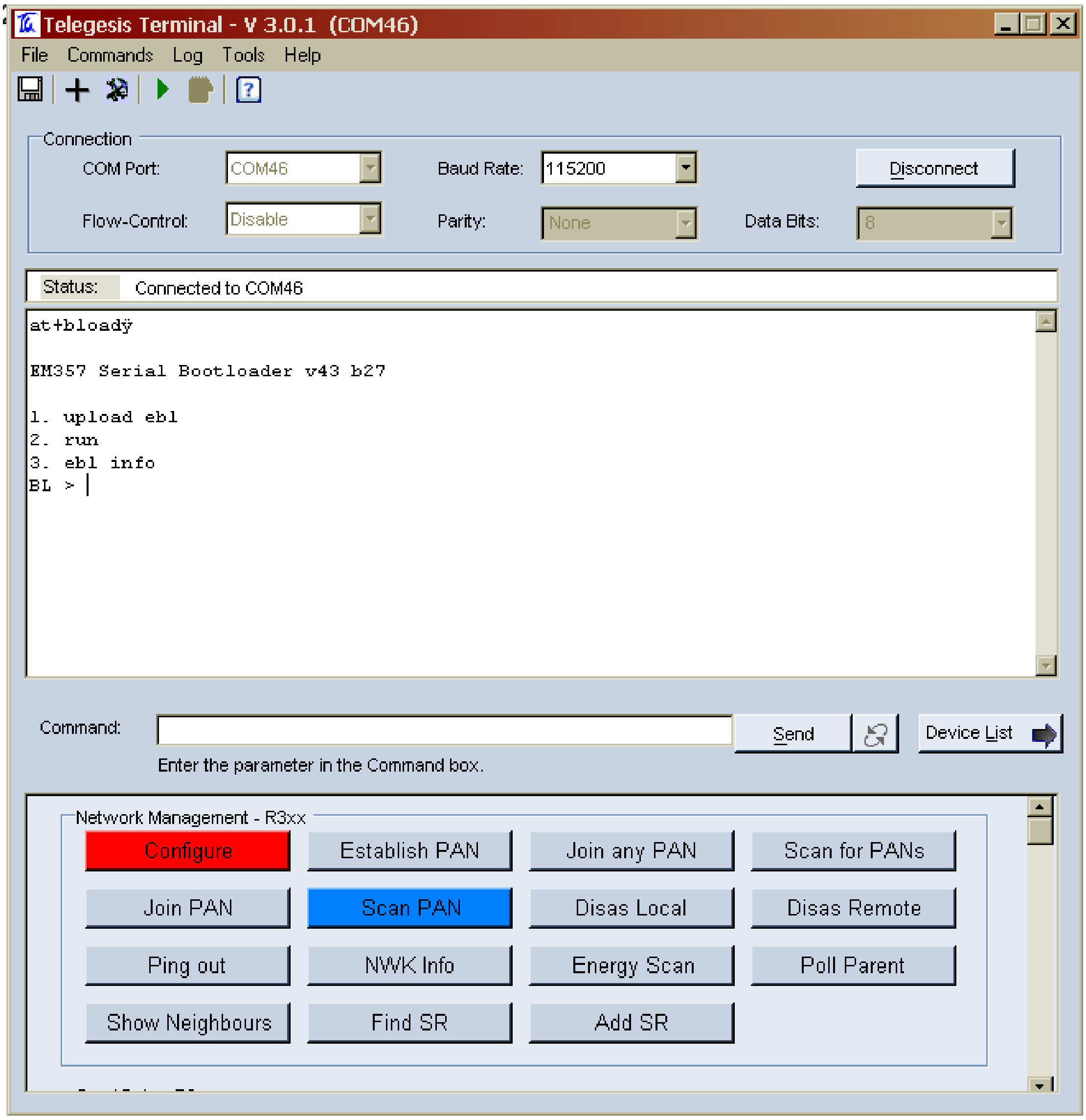
If required, the firmware of the ETRX3 modules can be upgraded serially.

7.1 Firmware Upgrades via Serial Port

In order to upgrade the firmware of the ETRX3 module using the serial bootloader, issue the “AT+BLOAD” command either by typing it in, or by pressing the respective button in the “Module Control” group of the Telegesis Terminal Application. Alternatively the button labeled “Bload” can be pressed on the development board whilst the reset button is pressed and released.

After entering the bootloader, the connection parameters need to be changed to 115200bps, 8 data bits, 1 stop bit, no parity, no flow control (providing that it is not already set to these values). This is achieved by pressing the ‘Disconnect’ button, changing the settings and then pressing the ‘Connect’ button (if only the connection speed needs to be changed disconnecting and reconnecting is not required).

After pressing ‘Enter’, the bootloader menu will be shown in the terminal window as shown in Figure 1.



COM Port: COM46
Baud Rate: 115200
Flow-Control: Disable
Parity: None
Data Bits: 8

Status: Connected to COM46

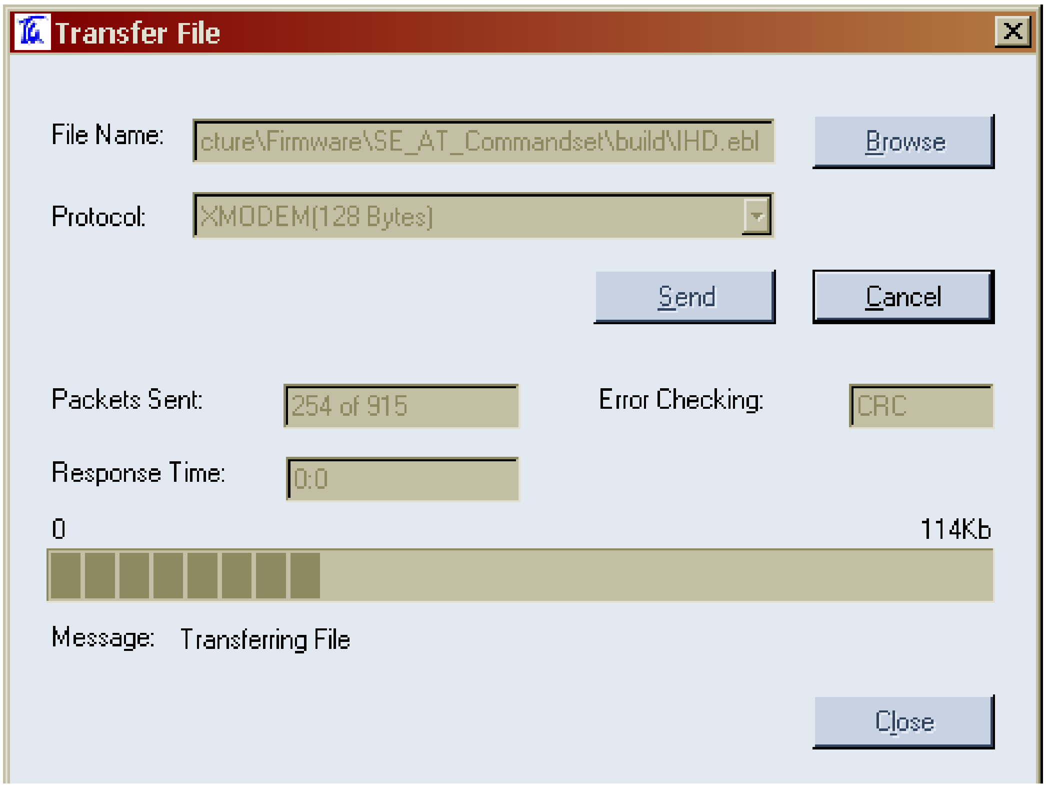
```
at+bloadÿ

EM357 Serial Bootloader v43 b27

1. upload ebl
2. run
3. ebl info
BL >
```

Enter the parameter in the Command box.

Figure 1 Options in Bootload Mode

Pressing ‘1’ initiates the upload of the new firmware and a number of ‘C’ characters will indicate that the ETRX3 is ready to receive data. Within 60 seconds, select Tools / Transfer File… and browse for the new firmware file.

Firmware files for the ETRX3 will be in the .ebl format. After checking that the protocol is set to XMODEM (128 Bytes), press the Send button and the new firmware will be downloaded as shown in Figure below.



Transfer File

File Name: cture\Firmware\SE_AT_Commandset\build\IHD.ebl

Protocol: XMODEM(128 Bytes)

Packets Sent: 254 of 915

Error Checking: CRC

Response Time: 0:0

Message: Transferring File

Figure 2 EBL Fill Transferred in Bootload Mode

When the transfer has been completed successfully, press Enter again in order to return to the bootloader menu (shown in figure 10) and option ‘2‘ to run the downloaded application software. If the application software has a baudrate other than 115200bps, this will need to be changed to the application baud rate as described above.

8 Trademarks

All trademarks, registered trademarks and products names are the sole property of their respective owners.

9 Disclaimer

Product and Company names and logos referenced may either be trademarks or registered trademarks of their respective companies. We reserve the right to make modifications and/or improvements without prior notification. All information is correct at time of issue. Telegesis (UK) Ltd. does not convey any license under its patent rights or assume any responsibility for the use of the described product.

10 Contact Information

Website:	www.telegesis.com
E-mail:	sales@telegesis.com

Telegesis (UK) Ltd.
Abbey Barn Business Centre
Abbey Barn Lane
High Wycombe
Bucks
HP10 9QQ
United Kingdom

Tel: +44 (0)1494 510199
Fax: +44 (0) 5603 436999