ZigBee OEM Module

ProBee-ZE20S-SESI
ProBee-ZE20S-SEID
ProBee-ZE20S-SEME

User Guide

Sena Technologies, Inc.

Rev 1.8

ProBee-ZE20S-SESI, ProBee-ZE20S-SEID, ProBee-ZE20S-SEME User Guide

Firmware version SESIv1.X
Firmware version SEIDv1.X
Firmware version SEMEv1.X

Copyright

Sena Technologies reserves the right to make changes and improvements to its product without providing notice.

Trademark

Notice to Users

When a system failure may cause serious consequences, protecting life and property against such consequences with a backup system or safety device is essential. The user agrees that protection against consequences resulting from system failure is the user's responsibility.

This device is not approved for life-support or medical systems.

Changes or modifications to this device not explicitly approved by Sena Technologies will void the user's authority to operate this device.

Technical Support

Sena Technologies, Inc.
210 Yangjae-dong Seocho-gu
Seoul 137-130, Korea
Tel: (+82-2) 576-7362
Fax: (+82-2) 573-7710
Email: support@sena.com
Website: http://www.sena.com

Revision	Date	Name	Description
V1.8	2013-08-22	Uin Burn	Initial Writing

Contents

1 Introduction

1.1 About This Document

This document provides an introduction on configuration and operation of the ZigBee Smart Energy(hereinafter referred to as the “SE”) Profile device such as the ProBee-ZE20S-SESI (Smart Energy Service Interface, hereinafter referred to as the “ProBeeSESI”), the ProBee-ZE20S-SEID(Smart Energy In-Premises Display, hereinafter referred to as the “ProBeeSEID”) and the ProBee-ZE20S-SEME(Smart Energy Metering Device, hereinafter referred to as the “ProBeeSEME”) which are based on the ProBee-ZE20S(hereinafter referred to as the “ZE20S”) ZigBee OEM module. This document assumes the user is using the ZE20S starter kit for evaluation hence the pictures and configuration examples shown in this document are all based on the ZE20S starter kit.

This document does not provide full detail of the hardware specifications. Please refer to the ProBee-ZE20S data sheet for further references on the hardware design.

1.2 ZE20S Overview

The ProBee-ZE20S is an all-in-one ZigBee OEM module with integrated ZigBee core and radio/antenna circuits together with high level software library. Using the ZE20S, OEM manufacturers can easily and cost-effectively integrate ZigBee functionality into target products in timely manner.

The ZE20S OEM module is designed to meet specific requirements of low cost and low power wireless applications such as Home Automation, Smart Energy, Health care, Building Automation and Telecom Services by adopting the ZigBee technology. The ZE20S is based on the Ember EM357 ZigBee core and certified ZigBee Alliance based on ZigBee 2007 and ZigBee Pro stack.

Despite of its low cost and low power consumptions, the ZE20S provides high performance and robust data transfer capability for wide range of applications. The ZE20S can transfer data at up to 250 kbps and reach up to 300 m (0.19 miles) outdoor line-of-sight.

Key Features

- Integrated 2.4GHz, IEEE 802.15.4-2003 compliant transceiver
- ZigBee 2007 / ZigBee Pro stack
- ZigBee core: Ember EM357
- Transmit Power: +8dBm (Boost mode)
- Receiver Sensitivity: -102dBm @1% BER (Boost mode)
- Supply Voltage: 2.1~3.6 VDC
- TX Current: 45mA @3.3V (max.)
- RX Current: 31mA @3.3V (max.)
- Sleep Current: ≤1uA

- Working Distance: 300 m (0.19 miles) @ 3.3V, +5dBi Dipole Antenna
- Various Antenna/Connector Options: Dipole 1/3/5dBi, U.FL, RPSMA, Chip
- UART Signals Support: UART_TXD/RXD, RTS/CTS, DTR/DSR
- 6 Analog Inputs
- 17 Digital Inputs/Outputs
- RoHS Compliant

Applications

- Advanced Metering Infrastructure
- Home Area Networks (HAN)
- Neighborhood Area Networks (NAN)
- Smart Energy
- Advanced lighting, entertainment and climate control systems
- Service-based monitoring, security and awareness systems
- Commercial Building Automation
- Climate & lighting control systems
- Industrial and Domestic Applications

1.3 ZigBee Smart Energy Profile

ZigBee Smart Energy Profile defines device descriptions and standard practices for Demand Response and Load Management "Smart Energy" applications needed in a Smart Energy based residential or light commercial environment. Installation scenarios range from a single home to an entire apartment complex. The key application domains included in this initial version are metering, pricing and demand response and load control applications. Other applications will be added in future versions.

1.4 Smart Energy Definitions

1.4.1 Cluster

Cluster is a container for one or more attributes and/or messages in a command structure. And Cluster is a related collection of attributes and commands, which together define a communications interface between two devices. The devices implement server and client sides of the interface respectively.

1.4.2 Attribute

Attribute is a data entity which represents a physical quantity or state. This data is communicated to other devices using commands.

1.4.3 Cluster identifier

Cluster identifier is a reference to the unique enumeration of clusters within a specific application profile. The cluster identifier is a 16-bit number unique within the scope of the application profile and identifies a specific cluster. Cluster identifiers are designated as inputs or outputs in the simple descriptor for use in creating a binding table.

1.4.4 Endpoint

Endpoint is a particular component within a unit. Each ZigBee device may support up to 240 such components. ZE20S has just 1 endpoint.

1.4.5 Client

Client is a cluster interface which is listed in the output cluster list of the simple descriptor on an endpoint. Typically this interface sends commands that manipulate the attributes on the corresponding server cluster.

1.4.6 Server

Server is a cluster interface which is listed in the input cluster list of the simple descriptor on an endpoint. Typically this interface supports all or most of the attributes of the cluster.

2 Getting Started

Creating a ZigBee network using the ZE20S modules consists of the following steps:

- Install the ZE20S module on the development board
- Create a ZigBee network by configuring a ZE20S module as the coordinator
- Configure ZE20S modules as routers. This step is optional since the coordinator works as a router too.
- Configure ZE20S modules as end-devices
- Transmit data between ZE20S devices

This chapter describes how to install the ZE20S modules on to the development boards and how to use terminal software programs to access the ZE20S modules using AT commands for configurations and operations. The rest of the steps including ZigBee network creation, coordinator/router/end-device configurations and data transmissions are described in the following chapters. The more detailed information on the starter kit and the configuration examples are also provided in the later chapter.

2.1 Hardware Installation

2.1.1 Development Board Layout

Figure 2-1 shows the layout of the development board. The development board provides the interface between the ZE20S and the external serial port or USB port together with input/output terminals for convenient tests.

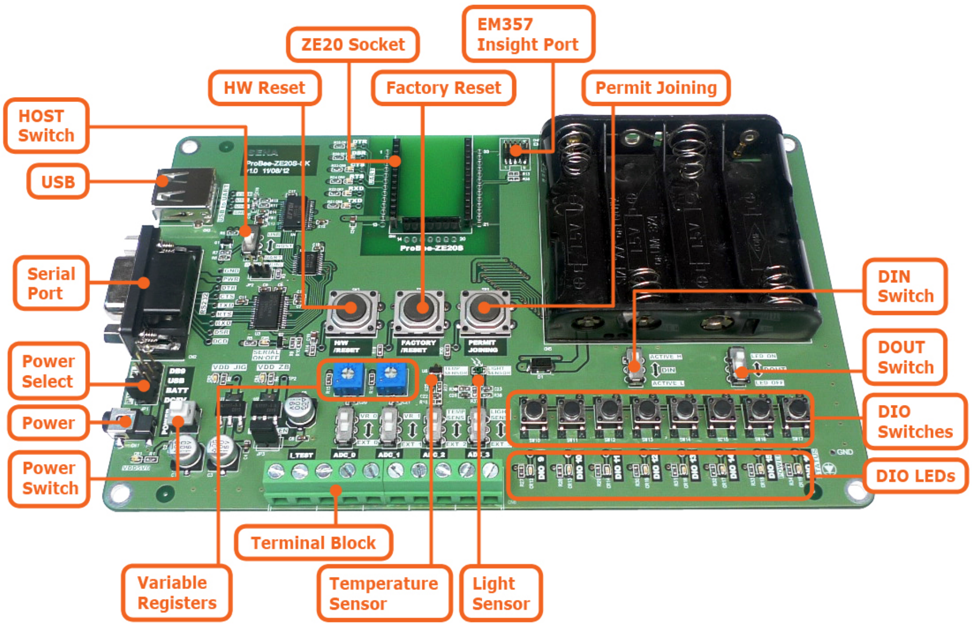


Figure 2-1 Development Board Layout

2.1.2 Antenna Installation

The ZE20S starter kit includes the several antennas for evaluation purposes. You can try different antennas for testing purpose and select the appropriate antenna for the actual applications. Figure 2-2 shows how to install the RP-SMA dipole antenna to the ZE20S module.

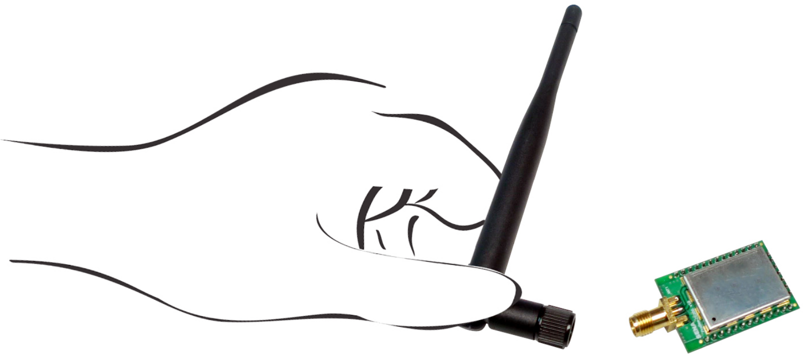


Figure 2-2 RP-SMA Dipole Antenna Installation

2.1.3 Module Installation on the Development Board

Once the antenna is installed on the ZE20S module, the ZE20S module can be installed on the development board.

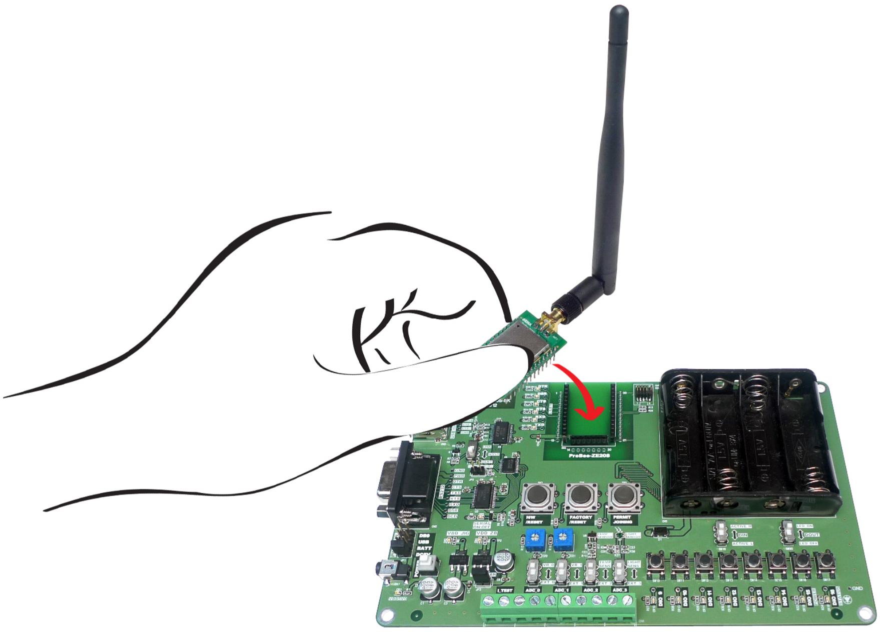


Figure 2-3 ZE20S Installation on the Development Board

2.1.4 Serial/USB Cable Installation

After the ZE20S is securely installed on the development board, please connect the included serial cable between the development board and your computer so AT commands can be sent from the computer to the ZE20S module. The serial port of the development board is DB9 female as a DTE terminal, and the serial cable included is a cross-over cable for DTE-DTE connection. To use the serial port, the HOST (SW2) switch should be set to UART.

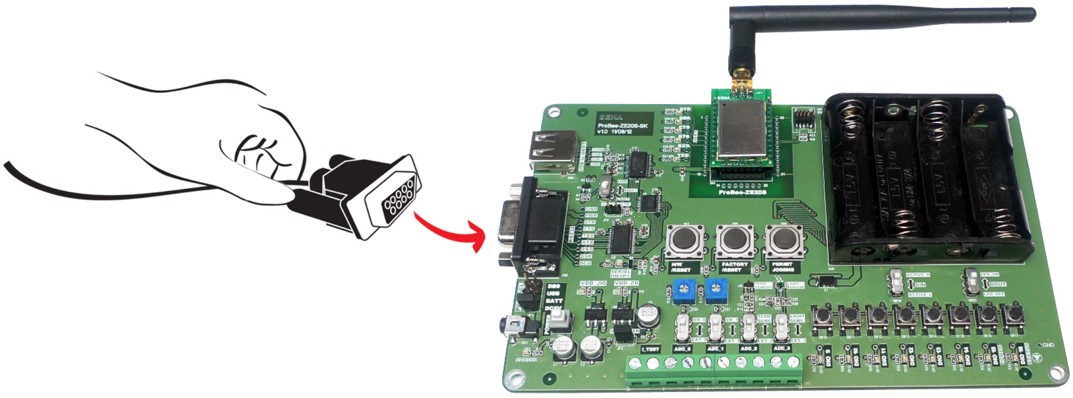


Figure 2-4 Serial Cable Installation

In case the serial port is not available for the computer, you can use the USB port of the development board instead. In this case, the internal USB-RS232 converter of the development board will convert the USB signal to the RS232 signal and vice versa. To use the USB port, the HOST (SW2) switch should be set to USB and the software driver for the USB in the CD should be installed on the computer. For the USB driver installation, please refer to the section 2.1.6.

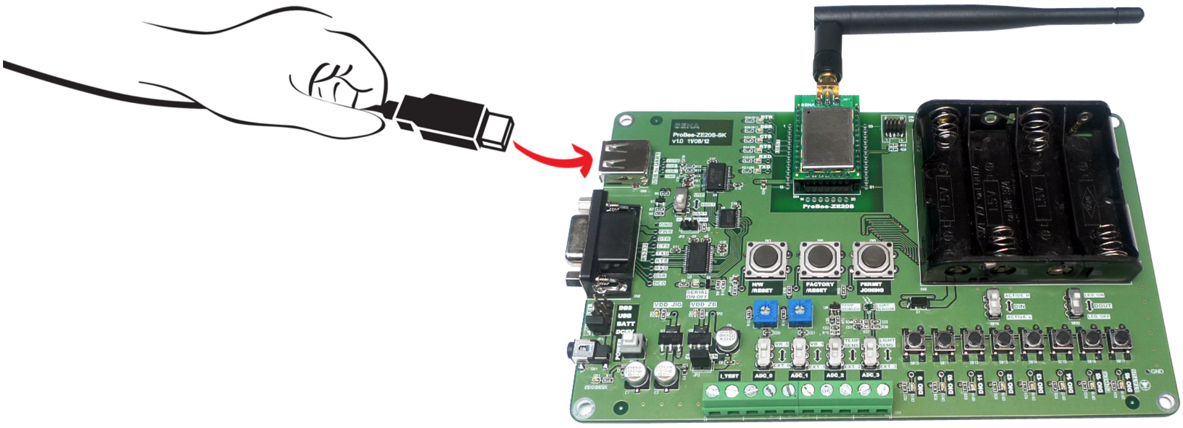


Figure 2-5 USB Cable Installation

2.1.5 Powering Development Board

Once all the installations are completed, please plug in the power connector using the included DC power adapter. The RS_PWR, USB_PWR and BATT_EN jumpers should be all OPEN to use the DC power adapter. Alternatively, the development board can be powered using batteries, or from the USB port or through the RS232 port pin 9. In these cases, the power source jumpers should be set accordingly. For details about the power source jumper settings, please refer to the section ***5.3.1 Power***.

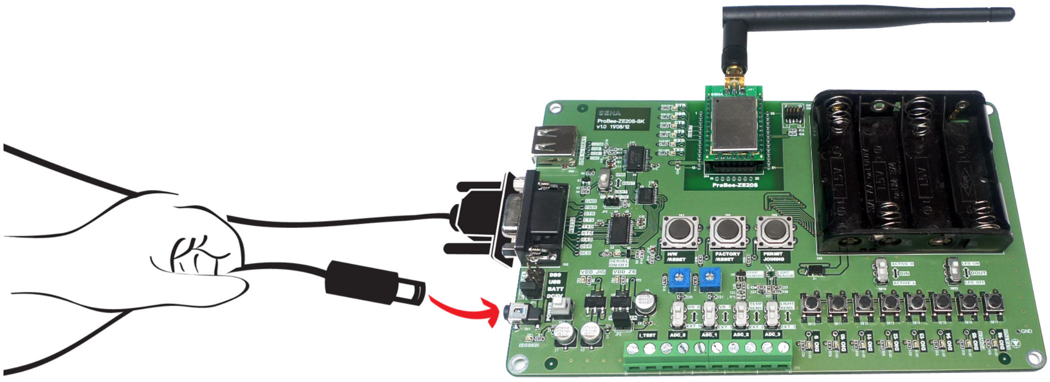
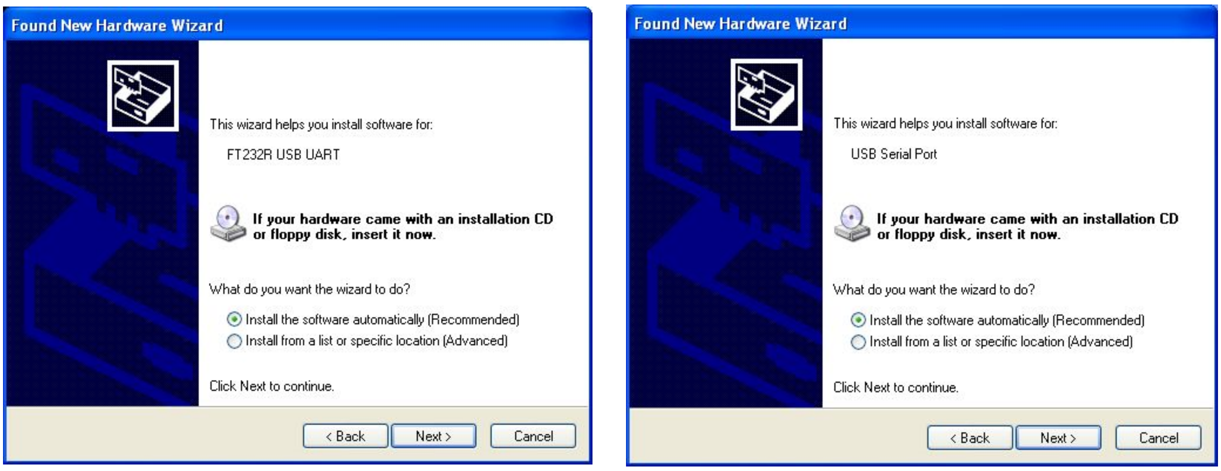


Figure 2-6 Plugging DC Power Adapter

2.1.6 USB Driver Installation

In case the USB interface is used for the communication between the development board and the computer, the corresponding software driver needs to be installed on the computer. Windows XP and later versions already include the USB driver. When Windows asks for the location of the USB driver, you can select the option to “install the software automatically” as shown in Figure 2-7. You can also choose to install the driver from the CD included if the driver is not installed correctly for some reason.



Found New Hardware Wizard

This wizard helps you install software for:

FT232R USB UART

If your hardware came with an installation CD or floppy disk, insert it now.

What do you want the wizard to do?

● Install the software automatically (Recommended)
○ Install from a list or specific location (Advanced)

Click Next to continue.

Found New Hardware Wizard

This wizard helps you install software for:

USB Serial Port

If your hardware came with an installation CD or floppy disk, insert it now.

What do you want the wizard to do?

● Install the software automatically (Recommended)
○ Install from a list or specific location (Advanced)

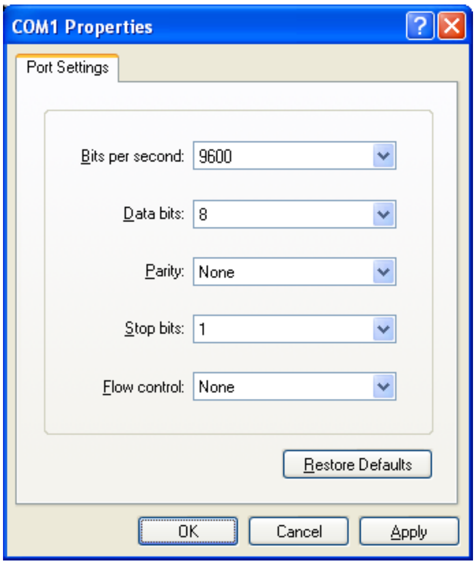
Click Next to continue.

Figure 2-7 USB Driver Installation

2.2 Using Terminal Software for AT Commands

Once the hardware installation is complete, the ZE20S unit is ready to communicate with the computer using the AT commands using a terminal emulation software. You can use any kind of terminal emulation software. This document assumes HyperTerminal is used. If you are using Windows Vista or 7, HyperTerminal is no longer included in the Windows so you need to download HyperTerminal from the Internet. For more details on HyperTerminal, please refer to Microsoft web page.

Once the HyperTerminal is installed, launch HyperTerminal and select the serial port that is connected to the ZE20S development board and enter the serial settings to use. The default serial settings of the ZE20S are **9600 bps, 8 data bit, none parity, 1 stop bit, none flow control**.



Bits per second: 9600

Data bits: 8

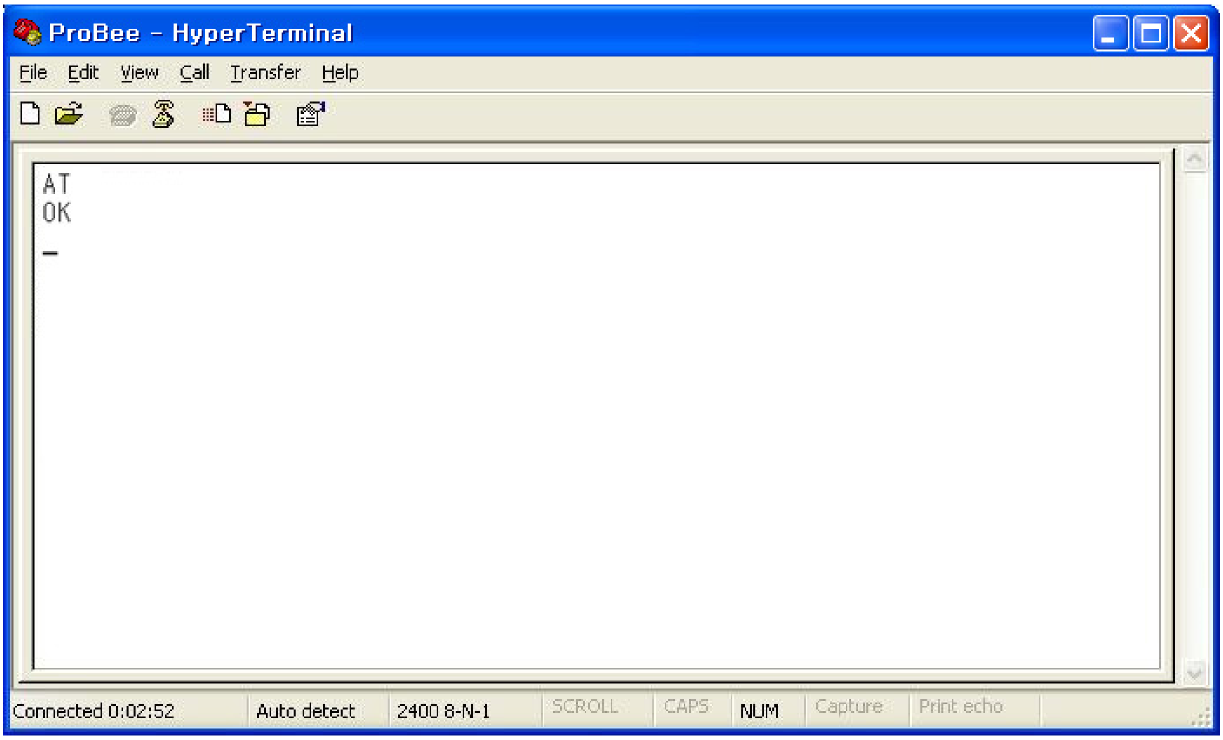
Parity: None

Stop bits: 1

Flow control: None

Figure 2-8 HyperTerminal Serial Settings

To display the AT commands that are being typed, you need to enable the local echo option on the HyperTerminal. To enable this option, Go to File->Properties->Settings->ASCII setup and select the "Echo typed characters locally" option. To verify the connection, type AT and press the Enter key. If the AT command is accepted by the ZE20S, OK string will be displayed on the screen.



```
AT
OK
_
```

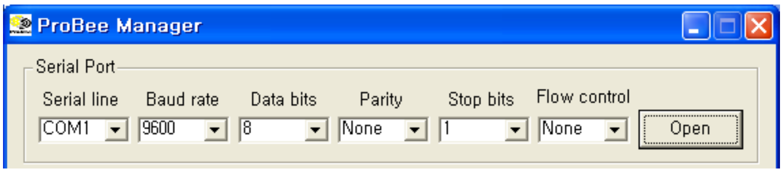
Figure 2-9 Testing AT Command on HyperTerminal

2.3 Using ProBee Manager for Configuration

It is possible to set or get the configurations of the ZE20S using ProBee Manager. This PC utility helps that a user can configure several setting values of local node and remote node easily without a terminal program. It is possible to export current configuration and import it to other nodes. It is also possible to restore to factory default configuration.

It provides users with functions to upgrade firmware of not only local node but also remote node.

2.3.1 Serial Port Open



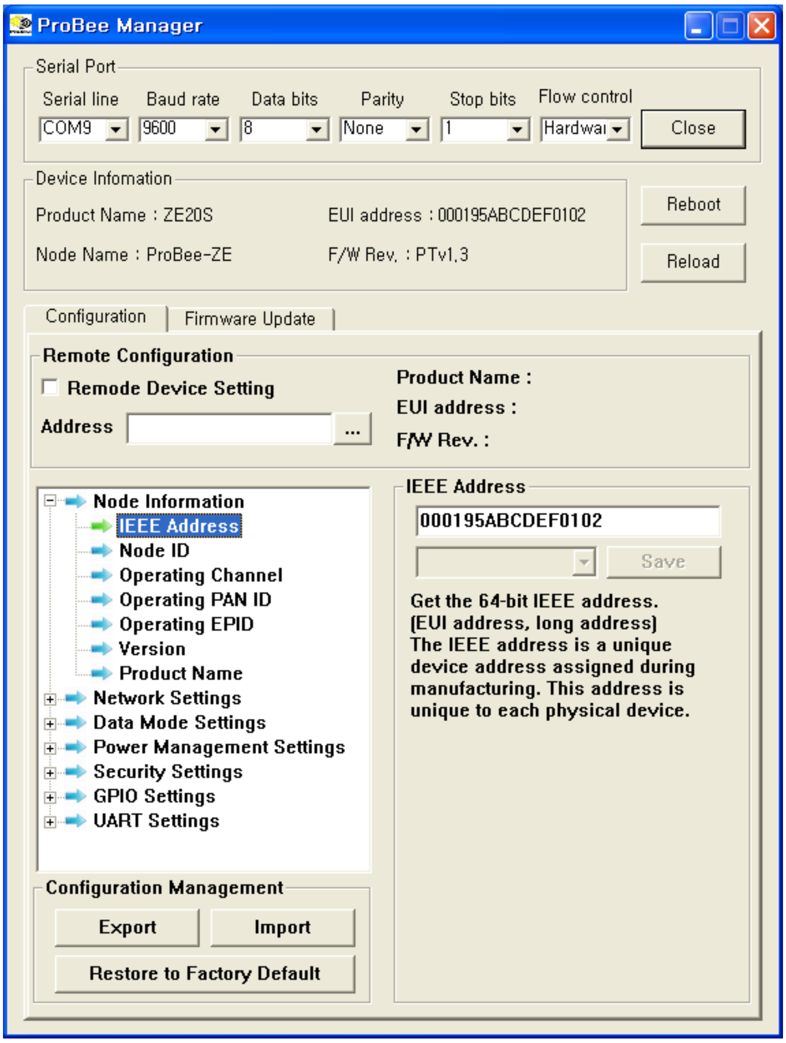
Serial line	Baud rate	Data bits	Parity	Stop bits	Flow control
COM1	9600	8	None	1	None

Figure 2-10 ProBee Manager Serial Port Open

To connect to local node through serial port, please follow the steps as below:

- Set 'Serial Port' properties corresponding to the UART configuration of local node.
- Click 'Open' button.

ProBee manager displays the main window like Figure 2-11 after connecting to local node.



Serial Port

Serial line	Baud rate	Data bits	Parity	Stop bits	Flow control
COM9	9600	8	None	1	Hardwai

Device Infomation

Product Name : ZE20S

EUI address : 000195ABCDEF0102

Node Name : ProBee-ZE

F/W Rev. : PTv1.3

Remote Configuration

□ Remode Device Setting

Address

Product Name :

EUI address :

F/W Rev. :

- Node Information
  - IEEE Address
  - Node ID
  - Operating Channel
  - Operating PAN ID
  - Operating EPID
  - Version
  - Product Name
- Network Settings
- Data Mode Settings
- Power Management Settings
- Security Settings
- GPIO Settings
- UART Settings

Configuration Management

IEEE Address

000195ABCDEF0102

Get the 64-bit IEEE address.
(EUI address, long address)
The IEEE address is a unique
device address assigned during
manufacturing. This address is
unique to each physical device.

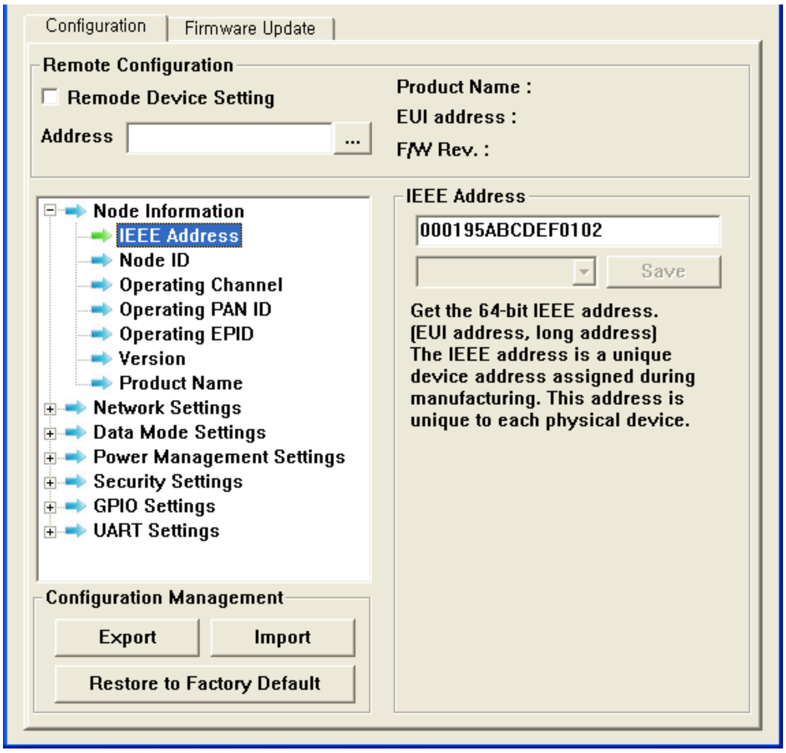
Figure 2-11 ProBee Manager Main Window

Clicking ‘Reboot’ button reboots the local node. Users can reload and refresh the recent configuration of the local node at ProBee manager by clicking ‘Reload’ button. ‘Configuration’ tab and ‘Firmware Update’ tab can help move to the ‘Configuration’ screen and the ‘Firmware Update’ screen.

2.3.2 Configuration

The Configuration Pane appears when the ‘Configuration’ tab is selected.

To change configuration, users need to find and modify the property and then apply changes. The properties are categorized according to the function. Selecting a category unfolds or folds the property list. The name, the current value and the description of the property are shown at the right of the configuration pane by selecting a property in the property list. It is possible to edit and save the current value of editable property. Whenever property is modified, it is necessary to save the change by clicking the ‘Save’ button in order not lose it moving to other properties. To apply changes, please reboot the device by clicking ‘Reboot’ button. The ‘Save’ button is disabled for read-only property.



000195ABCDEF0102

Get the 64-bit IEEE address.
[EUI address, long address]
The IEEE address is a unique device address assigned during manufacturing. This address is unique to each physical device.

Figure 2-12 ProBee Manager Configuration Pane

In Figure 2-12, the ‘Node Information’ category is selected and the ‘IEEE Address’ property is selected, so Its description is displayed. The ‘Save’ button is disabled because the ‘IEEE Address’ property is read-only.

The configurations of the remote nodes joining the network where the local node belongs can be modified, too. Because a sleepy end-device often enters sleep mode, to remote configuration from a sleepy end-device may not work.

To connect to local node through serial port, please follow the steps as below:

- Click the ‘…’ button at ‘Remote Configuration’ section.
- Scan the remote nodes joining the network and select the remote node to configure, that results in displaying the remote node in the ‘Address’ edit box.
- Check the ‘Remote Device Setting’ check box and the caption of the ‘Restore to Factory Default’ button at the ‘Configuration Management’ section becomes ‘Reboot’ that makes the remote node reboot and configuration of the selected node is displayed.
- Select and modify the property.
- Click the ‘Save’ button.

- Click the ‘Reboot’ button whose caption used to be ‘Restore to Factory Default’ to apply changes.

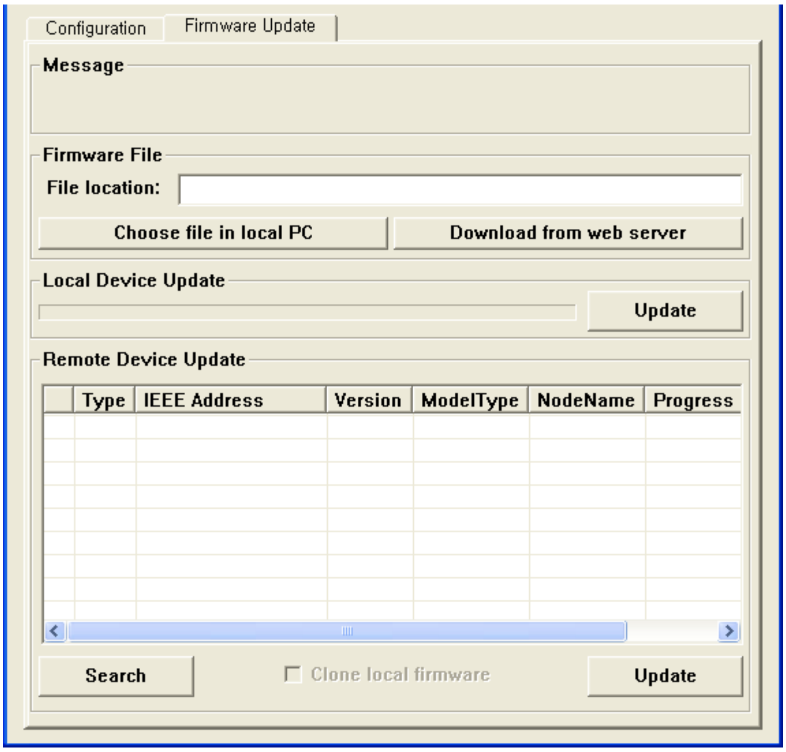
To export the configuration, please follow the steps as below:

- Click the ‘Export’ button at the ‘Configuration Management’ section and the ‘Save As’ dialog pops up.
- Select the path and enter the file name to store the configuration at.
- Click the ‘Save’ button’ and the configuration is exported to the designated file.

Exported file can be used at other devices by importing them. The configuration of remote node can be exported and imported by checking the ‘Remote Device Setting’ check box.

Users can reset to factory default by clicking the ‘Restore to Factory Default’ at the ‘Configuration Management’ section.

2.3.3 Firmware Update



	Type	IEEE Address	Version	ModelType	NodeName	Progress

Figure 2-13 ProBee Manager Firmware Update Pane

The Firmware Update Pane appears when the ‘Firmware Update’ tab is selected. The ‘Message’ section displays informative message about the progress of updating firmware.

To update firmware at local node, please follow the steps as below:

- Select a firmware file to update at the ‘Firmware Update’ section. The ‘Choose file in local PC’ button pops up the dialog where users can select the firmware file at local PC and the ‘Download from web server’ button helps users download the firmware from web server.
- Click the ‘Update’ button at the ‘Device Update’ section to update the local node. The progress rate is expressed at the ‘Local Device Update’ section and informative messages are shown at ‘Message’ section.

ProBee Manager also supports to update the firmware of remote nodes which join the network where local node is. Because a sleepy end-device often enters sleep mode, to update remote node from a sleepy end-device may not work.

To update firmware at remote node, please follow the steps as below:

- Select a firmware file.
- Click the ‘Search’ button, that scans the remote nodes joining the network where local node belongs.
- Select the remote nodes to update in the remote nodes list.
- Click the ‘Update’ button at the ‘Remote Device Update’ section.

3 ZigBee Network Configuration

A ZigBee Network consists of a coordinator, routers and end-devices. A minimal ZigBee network consists of one coordinator and multiple end-devices which directly connect to the coordinator. For larger ZigBee networks, routers are required to provide redundant routings to form mesh networks.

- **ZigBee Coordinator**: The coordinator forms the root of the network tree and might bridge to other networks. There is exactly one ZigBee coordinator in each network since the coordinator starts the ZigBee network itself. It also stores information about the network, including acting as the Trust Center & repository for security keys.
- **ZigBee Router**: The routers are able to route messages between other devices. Since the coordinator provides the routing functionality too, the routers are optional to form a ZigBee network.
- **ZigBee End-Device**: The end-device contains just enough functionality to talk to the parent node which is either the coordinator or a router. It cannot relay data from other devices.
- **ZigBee Sleepy End-Device**: The sleepy end-device is the same with normal end-device except it can stay in the sleep mode much longer. This relationship allows the end-device to sleep significant amount of the time and achieve the long battery life.

The ZE20S can be configured to work either as a coordinator, a router, an end-device or a sleepy end-device. Once the operation mode of the ZE20S is selected, corresponding parameters need to be set up appropriately. This chapter explains how to configure the ZE20S operation mode and other parameters related to the ZigBee network itself.

Table 3-1 ZigBee Node Types

Node Type	**Recommendation when to use ZE20S**
Coordinator	It should be selected when it is first time to establish the network.
Router	It is recommended when there is a network already and it needs to be expanded. A router is suitable for fast and stable data transmissions and receptions.
End-Device	It is recommended when it does not need to expand the network anymore and there are enough nodes as parents. An end-device consumes less power than a router.
Sleepy End-Device	It is recommended when the data transmissions occur rarely and in very small quantities. A sleepy end-device can lose data if sleep duration is too long, but it is possible to consume extremely low power as low as 2uA.

3.1 ZigBee Security

Security is a major concern in the ZigBee architecture. Although ZigBee uses the basic security elements in IEEE 802.15.4 (e.g., AES encryption & CCM security modes), the ZE20S increases the security level with:

- 128 bit AES encryption algorithms
- Strong, NIST-approved security
- Defined Key Types (Link, Network)
- Defined Key setup and maintenance
- CCM (Unified/Simpler mode of operation)
- Trust Centers

3.1.1 Security Level

In order to use security in the application, a user should set security level using the AT command below. Set 1 defines both Authentication and Encryption at the network layer. Set 0 defines no security. Disabling security in the application is not ZigBee compliant. All nodes in the network should have the same security level setup. The security level is configured using the AT command below.

AT+SECURITY=n or AT+SE=n, where n is encryption level. (0=disable, 1=enable)

ZigBee Smart Energy Network is required to high level security and use Pre-configured Link Key with Certificate-Based Key Establishment solution in each device.

3.1.2 CBKE (Certificate-Based Key Establishment)

The CBKE (Certificate-Based Key-Establishment) solution uses public-key technology with digital certificates and root keys. Each device has a private key and a digital certificate that is signed by a Certificate Authority (CA). The digital certificate includes:

- Reconstruction data for the device's public key
- The device's extended 64-bit IEEE address
- Profile specific information (e.g., the device class, network id, object type, validity date, etc.).

Certificates provide a mechanism for cryptographically binding a public key to a device's identity and characteristics. Trust for a CBKE solution is established by provisioning a CA root key and a digital certificate to each device. A CA root key is the public key paired with the CA's private key. A CA uses its private key to sign digital certificates and the CA root key is used to verify these signatures. The trustworthiness of a public key is confirmed by verifying the CA's signature of the digital certificate. Certificates can be issued either by the device manufacturer, the device distributor, or the end customer. For example, in practical situations, the CA may be a computer (with appropriate key management software) that is kept physically secure at the end customer's facility or by a third-party. At the end of successful completion of the CBKE protocol the following security services are offered:

- Both devices share a secret link key
- Implicit Key Authentication: Both devices know with whom they share this link key
- Key Confirmation: Each device knows that the other device actually has computed the key correctly
- No Unilateral Key Control: No device has complete control over the shared link key that is established
- Perfect Forward Secrecy: if the private99 key gets compromised none of future and past communications are exposed
- Known Key Security resilience: Each shared link key created per session is unique

3.1.3 Link Key

The trust center (normally coordinator in network) requests link key before a device is trying to join network. If link key the device has is wrong, joining process will be denied. Hence, all nodes in the network should have the valid link key setup before joining the network. Before joining the network, link key is generated by CBKE Data Token and Install Code. CBKE Data Token consist of 'Device Implicit Cert', 'CA Public Key', and 'Device Private Key'. The CBKE Data Token and Install Code is configured using the AT command below.

AT+IMPLICIT= xxx…xxx or AT+IC = xxx…xxx, where xxx…xxx is 48-byte hexadecimal CBKE Data Token(Device Implicit Cert).

AT+PUBLICKEY= xxx…xxx or AT+PK = xxx…xxx, where xxx…xxx is 43-byte hexadecimal CBKE Data Token(CA Public Key & Device Private Key).

AT+INSTALLCODE= xxx…xxx or AT+IC = xxx…xxx, where xxx…xxx is 6/8/12/16-byte hexadecimal Install Code.

3.1.4 Network Key

It is the network-wide key used to secure transmissions at the Network Layer. The network key encrypts all transmissions at the Network Layer. The network key only needs to be set up for the coordinator.

AT+NWKKEY=xxx…xxx or AT+NK=xxx…xxx, where xxx…xxx is 128-bit hexadecimal network key. Default is FFF…FF.

3.1.5 Update Key

A trust center can update the Network Key to help minimize the risk associated with a particular instance of the network key being compromised. Key updates are broadcast by the trust center throughout the network, encrypted using the current network key. Devices that hear the broadcast will not immediately use the key, but simply store it. Later, a key switch is broadcast by the trust center to tell all nodes to start using the new key. At a minimum, the trust center should allow adequate time for the broadcast of the new key to propagate throughout the network before switching.

AT+UPDATEKEY<CR> or AT+UK<CR>

3.2 Configuring CBKE Data Token

Generated link key is required to support Smart Energy security, which is based on CBKE using Elliptic-Curve Cryptography (ECC). To generate valid link key, you should obtain correct certificate (Implicit Cert, Public Key, and Device Private Key) and input that to CBKE Data Token in ZE20S node. Here is the method for obtaining test certificate. You can get test certificates by following step:

1. Register for an account (for test certificate generation) at:
http://www.certicom.com/index.php/gencertregister.
This registration is valid for a limited time.

2. Log in to the test certificate generation site with your account login at:
http://www.certicom.com/index.php/devicelogin.

3. Go to the certificate generation page:
http://www.certicom.com/index.php/smartenergydevicecertificateservice.

4. In the Subject ID field, enter the MAC address (long address) of the target device where the certificate will reside (MSB order, such as 000195...). This should be same as your target node's MAC address, but if the address differs, the target device will not work as Smart Energy node. You can get the MAC address of your ZE20S node by AT+LONGADDR.

Example)

```
AT+LONGADDR<CR>         # Get MAC address (long address) of current node.
000195AABBCC0003<CR>    # Input this value to subject ID field.
OK<CR>                  # Response
```

5. In the Profile Attribute Data field, enter any custom hexadecimal you want associated with this device. (The recommended format is profile ID [0109] + cert type [1 for test] + 16-bit ZigBee Manufacturer Code + 15-byte optional customer data)

Example)
0109108301234567890A

6. Leave the Public Key field empty.

7. Click the Generate button. You can see resulting page as below. Device Public Key will not be used.

Results!

CA Pub Key:	0200fde8a7f3d1084224962a4e7c54e69ac3f04da6b8
Device Implicit Cert:	03022cd8b45f0f21676e4184470d34c3626fc10485741 122334455667788544553545345434111223344556 6 77889900
Device Private Key:	03b546c41cad5b6c4407144149362c478f844b0d6d
Device Public Key:	03075012564096b1b93d6fd5d450401410fa50d9994b

8. Copy ‘Device Implicit Cert’ value and input to ZE20S node.

Example)

If resulting page shows:

```
Device Implicit Cert:
03022cd8b45f0f21676e4184470d34c3626fc10485741122334455667788544553545345434111
2233445566778899 00
```

Copy that value and input to ZE20S node by AT+IMPLICIT command:

```
AT+IMPLICIT=03022cd8b45f0f21676e4184470d34c3626fc10485741122334455667788544553
54534543411122334455667788 9900<CR>

CBKE DATA TOKEN IS EMPTY<CR>

DEVICE IMPLICIT CERT(BUFFER):
03022CD8B45F0F21676E4184470D34C3626FC10485741122334455667788544553545345434111
2233445566778899 00<CR>

CA PUBLIC KEY & DEVICE PRIVATE KEY(BUFFER):
000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000<CR>

OK<CR>                         # Response
```

9. Copy ‘CA Pub Key’ + ‘Device private key’ value and input to ZE20S node.

Example)

If resulting page shows:

```
CA Pub Key:  0200fde8a7f3d1084224962a4e7c54e69ac3f04da6b8

Device Private Key: 03b546c41cad5b6c4407144149362c478f844b0d6d
```

Copy that value and input to ZE20S node by AT+PUBLICKEY command:

```
AT+PUBLICKEY=0200fde8a7f3d1084224962a4e7c54e69ac3f04da6b803b546c41cad5b6c44071
44149362c478f844b0d6d<CR>
                  # Input CA Pub Key (22 byte) + Device Private Key (21 byte)
CBKE DATA TOKEN IS EMPTY<CR>
DEVICE IMPLICIT CERT(BUFFER):
03022CD8B45F0F21676E4184470D34C3626FC104857411223344556677885445535453454341112
2334455667788 9900<CR>
CA PUBLIC KEY & DEVICE PRIVATE KEY(BUFFER):
0200fde8a7f3d1084224962a4e7c54e69ac3f04da6b803b546c41cad5b6c4407144149362c478f
844b0d6d <CR>
OK<CR>            # Response
```

10. Lock the CBKE Data Token by AT+SETCBKEKEY.

Example)

```
AT+SETCBKEKEY<CR>
CBKE DATA TOKEN IS LOCKED<CR>             # CBKE Data Token is locked now
DEVICE IMPLICIT CERT:
03022CD8B45F0F21676E4184470D34C3626FC10485741122334455667788544553545345434111
22334455667788 9900<CR>
CA PUBLIC KEY & DEVICE PRIVATE KEY:
0200fde8a7f3d1084224962a4e7c54e69ac3f04da6b803b546c41cad5b6c4407144149362c478f
844b0d6d<CR>
```

Once you lock CBKE Data Token, you cannot change CBKE Data Token again.
You can check if it is good or bad by AT+INFO.

Example) Good CBKE Data Token

```
AT+INFO<CR>
Security Level: 05, Security Profile: SE Full
SE Security Info: RealEcc, RealCbke, GoodCert        # Good CBKE Data Token
End-point count: 1
…
```

Under following situations, you can see ‘BadCert’ message in the result of AT+INFO.

- Input invalid Device Implicit Cert value at AT+IMPLICIT
- Input invalid CA Pub Key + Device private key value at AT+PUBLICKEY
- CBKE Data Token value is not generated with MAC address of current node

Example) Bad CBKE Data Token

```
AT+INFO<CR>
Security Level: 05, Security Profile: SE Full
SE Security Info: RealEcc, RealCbke, BadCert          # Bad CBKE Data Token
End-point count: 1
…
```

Note: Once the node is locked with BAD CBKE Data Token, it cannot work as Smart Energy node and its CBKE Data Token cannot be changed and erased. There is no way to change or erase CBKE Data Token. Even Factory Reset does not erase its CBKE Data Token. Please be sure to lock the CBKE Data Token with valid certificate.

3.3 Configuring Install Code

User can set the 'Install Code' for joining a ZE20S device to the SE network. Install Code is used to generate Pre-configured Link Key. The Pre-configured Link Key is derived from the Install Code using the MMO hash function. Install code is optional. While the Trust Center (coordinator) can communicate with any node other than the Trust Center, non-Trust Center nodes with common Install Code in a Smart Energy Network can communicate with each other. In ZE20S node, user can set Install Code by AT+INSTALLCODE. Install Code must be 6/8/12/16 byte hexadecimal.

Example)

```
AT+INSTALLCODE=AABBCCDDEEFF<CR>     # Install Code must be 6/8/12/16 byte
                                    # hexadecimal
OK<CR>                              # Response

AT+INSTALLCODE<CR>
INSTALL CODE: AABBCCDDEEFF<CR>      # Install Code locked
OK<CR>                              # Response
```

Note: Once the Install Code is set, it cannot be changed and erased. There is no way to change or erase Install Code. Even Factory Reset does not erase Install Code. Please be sure to input the Install Code with care.

3.4 Setting up ZE20S as a Coordinator

A coordinator relays messages and can act as a parent to other nodes. Every personal area network (PAN) must be started by a node acting as the coordinator. In general, this device is always powered on. To set up the ZE20S to work as the coordinator, please follow the steps as below

- Configure the CBKE Data Token
- Configure the ZE20S node type as the coordinator
- Optionally, select the channel mask. Otherwise, the ZE20S will use the default channel mask.

- Optionally, set up the PAN ID and/or the extended PAN ID. Otherwise, the ZE20S will generate the PAN ID and/or the extended PAN ID automatically.
- Permit joining when other router or end-device tries to join the ZigBee network.
- Add Pre-configured Link Key to Key Table for Smart Energy Security.

3.4.1 CBKE Data Token Configuration

Before operating ZE20S module as Smart Energy node, user should set CBKE certificate to CBKE Data Token for using Smart Energy security. Please refer to the section ***3.2 Configuring CBKE Data Token*** for getting certificate and setting CBKE Data Token.

3.4.2 Node Type Selection for Coordinator

The ZE20S module can be configured as a ZigBee coordinator by the AT command as below:

AT+NODETYPE=n or AT+NT=n
, where n is 0 = none, 1 = coordinator, 2 = router, 3 = end-device, 4 = sleepy end-device

After the operation mode is selected, ATZ command should be performed to reset the unit to have the change applied.

Example)

```
AT+NODETYPE=1<CR>              # Set node type as the coordinator
OK<CR>                         # Response
ATZ<CR>                        # Apply the change
OK<CR>                         # Response
```

3.4.3 Channel Mask Configuration for Coordinator

When the ZigBee coordinator starts, it scans predefined frequency channels to find the best frequency channel to use without causing interference with other wireless devices. The user is able to specify the channels to use with the AT command as below:

AT+CHMASK=n or AT+CM=n
, where n = channel mask to use. Default = 0x0318C800

This value has a 32-bit hexadecimal unsigned integer. Each bit represents whether a channel will be used or not. For example, to use a channel between 12 and 15, the 12, 13, 14 and 15 bits of the channel mask should be 1 and all other bits should be 0. The default setting is 0x0318C800 which means that all the channels from 11 to 26 can be used. The ZE20S supports only the channels between 11 and 26, so 0~10 bits and 27~31 bits should be 0. Table 3-2 shows the bit mask for each channel.

Table 3-2 Channel Bitmask

Channel	11*	12	13	14*	15*	16	17	18
Bit mask	00000800	00001000	00002000	00004000	00008000	00010000	00020000	00040000
Channel	19*	20*	21	22	23	24*	25*	26
Bit mask	00080000	00100000	00200000	00400000	00800000	01000000	02000000	04000000

* Preferred Channel Usage in Smart Energy Network (Default channel)

3.4.4 PAN ID Configuration for Coordinator

Every ZigBee network should be assigned a PAN ID to identify the network. If the PAN ID is 0000, the ZE20S will generate a random 16-bit PAN ID to start the ZigBee network. Otherwise, the user can enter the PAN ID using the AT command as below:

AT+PANID=xxxx or AT+PI=xxxx
, where xxxx is PAN ID to use. For example, 12AB. Default setting is 0000

When choosing a PAN ID, the user needs to make sure that there is no ZigBee network using the same PAN ID. If the PAN ID is already used by another ZigBee network, the PAN ID is unavailable and the user should choose a new PAN ID. In order to avoid this, PAN ID auto-generation is strongly recommended.

The user can also assign the 64-bit extended PAN ID to the ZE20S using the AT command below. If the extended PAN ID is 0000000000000000, the ZE20S uses its IEEE address as the extended PAN ID. The extended PAN ID cannot be shared with other ZigBee networks, either.

AT+EPID=xxx···xxx or AT+EI= xxx···xxx
, where xxx···xxx is EPID to use. For example, 0123456789ABCDEF. Default setting is 000···000.

3.4.5 Permit Joining

When a router or an end-device tries to join the ZigBee network, there should be at least one coordinator or router which is permitting joining. If there is no router in the ZigBee network yet, the coordinator should permit joining. The ZE20S provides three ways to permit joining are as follows:

- Permanently permitting by configuration
- Transiently permitting by AT command
- Transiently permitting by digital input signal

If joining is permanently permitted, any device can join anytime. Since this will allow even an unidentified device to join the network anytime, this mode should be selected with extra caution. The AT command to permit joining permanently is as follows:

AT+PERMIT=255 or AT+PJ=255

The user can permit joining for a moment at need. The AT command to permit joining is

AT+PERMIT=n or AT+PJ=n

, where n is the duration to permit in seconds ranging 1~254 seconds.

The user can also configure the ZE20S to permit joining when the digital input GPIO0 is active. The AT command to enable this functionality is as follows:

AT+GPIO0=5 or AT+IO0=5

When the ZE20S detects falling edge of the GPIO0, the ZE20S permits joining for the duration that is configured in S-register 22 (S22). The AT command to change the duration is as follows:

ATS22=n

, where n is the duration to permit in seconds ranging 1~254 seconds. Default is 255.

S22 value determines the time that the device will allow other devices to join to it. If set to 255, the device will always permit other nodes joining. If the user reboots the coordinator (or a router), the device allows other nodes to join the network for the period defined by S-register 22.

Note: When a device that has joined a network reboots, the device is able to join the network regardless of permitting. But, after the device has joined other network, rejoining the previous network should be permitted by the coordinator or a router.

3.4.6 Add Pre-configured Link Key to Key Table

In ZE20S Smart Energy network, Coordinator acts as Trust Center. To permit joining other nodes, it needs to add the long address and Pre-configured Link Key of the joining nodes to its Key Table. The AT command to add Pre-configured Link Key to Key Table is as follows:

AT+KEYTABLE=n,long_address,link_key

, where n is the number of key table ranging 0~5, long_address is 64-bit IEEE address of joining node and link_key is 128-bit pre-configured key of joining node.

AT+KEYTABLE is also used for getting Pre-configured Link Key in joining node. If you want to get Pre-configured Link Key of joining node, you should input AT+KEYTABLE without parameters.

Example) Use AT+KEYTABLE in router or end-device to get Pre-configured Link Key

```
AT+KEYTABLE<CR>               # Input AT+KEYTABLE for getting Link Key
NETWORK KEY: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

LINK KEY TABLE
#|IEEE_ADDRESS    |TYPE|AUTH|KEY
*|0000000000000000|LINK|NO  |1A51778BAF98BEFAD6B41215F402C979
                              # KEY(1A51778BAF98BEFAD6B41215F402C979) value is
                              # Pre-configured Link Key of current node
1/6 ENTRIES USED
OK<CR>                        # Response

AT+LONGADDR<CR>               # Get long address of joining node
0001951122334455<CR>
OK<CR>                        # Response
```

Example) Use AT+KEYTABLE in Coordinator to add Pre-configured Link Key

```
AT+KEYTABLE=0,0001951122334455,1A51778BAF98BEFAD6B41215F402C979<CR>
                              # Add Pre-configured Link Key to Key Table
OK<CR>                        # Response
```

Then the joining node can allow to join to network.

3.5 Setting up ZE20S as a Router

A router is a full-function routing device that relays messages and can act as a parent to other nodes. These devices must be always powered on.

To configure the ZE20S as a ZigBee router, please follow the steps below.

- Configure the CBKE Data Token
- Optionally, configure the Install Code
- Configure the ZE20S node type as a router.
- Optionally, select the channel mask. Otherwise, the ZE20S will use the default channel mask.
- Optionally, set up the PAN ID and/or the extended PAN ID. Otherwise, the ZE20S will scan and join the ZigBee network found automatically.
- Permit joining when other router or end-device tries to join the ZigBee network.

3.5.1 CBKE Data Token Configuration

Before operating ZE20S module as Smart Energy node, user should set CBKE certificate to CBKE Data

Token for using Smart Energy security. Please refer to the section ***3.2 Configuring CBKE Data Token*** for getting certificate and setting CBKE Data Token.

3.5.2 Install Code Configuration

Router and end-device can set Install Code optionally for enhanced Smart Energy security. Install Code is used to generate Pre-configured Link Key. Nodes other than Trust Center in a Smart Energy network must have same Install Code if they are able to communicate with each other. Please refer to the section ***3.3 Configuring Install Code*** for setting Install Code.

3.5.3 Node Type Selection for Router

The ZE20S module can be configured as a ZigBee router by the AT command as below:

AT+NODETYPE=n or AT+NT=n

, where n is 0 = none, 1 = coordinator, 2 = router, 3 = end-device, 4 = sleepy end-device

After the operation mode is selected, ATZ command should be performed to reset the unit to have the change applied.

Example)

```
AT+NODETYPE=2<CR>             # Select router mode
OK<CR>                        # Response
ATZ<CR>                       # Apply the change
OK<CR>                        # Response
```

3.5.4 Channel Mask Configuration for Router

When the ZE20S as a ZigBee router or a ZigBee end-device tries to join, it scans all the channels configured by channel mask to find the ZigBee network to join. The channel mask of router or end-device should include the channel on which the coordinator is operating. On the assumption that the coordinator of the network to join is operating on channel 12, the router or end-device trying to join should have a channel mask of which 12th bit is 1.

3.5.5 PAN ID Configuration for Router

The PAN ID of a ZigBee router or a ZigBee end-device should be the same with the PAN ID of the ZigBee coordinator. If the coordinator is using a randomly generated PAN ID, the user can get the operating PAN ID by running the AT command below on the coordinator:

AT+OPPANID or AT+OI

If the PAN ID of a ZigBee router or a ZigBee end-device is 0000, the PAN ID is ignored. After checking the PAN ID, the ZE20S trying to join compares the extended PAN ID to make sure whether the network is the right network to join. If the extended PAN ID is the same with its configured extended PAN ID, it will try to join the network. Otherwise, it will try to find another network. The user can get the operating extended PAN ID by running the AT command below on the coordinator:

AT+OPEPID or AT+OE

If the extended PAN ID of a ZigBee router or a ZigBee end-device is 0000000000000000, the extended PAN ID is ignored

3.5.6 Permit Joining

The ZigBee router is configured and works the same way with the coordinator to permit joining of other nodes. Please refer to the section ***3.4.5 Permit Joining*** for the configuration.

3.6 Setting up ZE20S as an End-Device

An end-device is a leaf node which communicates only with its parent node and will not relay messages directly. To configure the ZE20S as a ZigBee end-device, please follow the steps below.

- Configure the CBKE Data Token
- Optionally, configure the Install Code
- Configure the ZE20S node type as an end-device
- Optionally, select the channel mask. Otherwise, the ZE20S will use the default channel mask.
- Optionally, set up the PAN ID and/or the extended PAN ID. Otherwise, the ZE20S will scan and join the ZigBee network found automatically.
- Configure the poll timeout and the poll period

Please refer to the section ***3.5 Setting up ZE20S as a Router*** for the AT commands regarding the CBKE Data Token configuration, the Install Code configuration, the node type selection, the channel mask configuration and the PAN ID/Extended PAN ID configuration.

When an end-device joins the network, the power LED blinks periodically if GPIO15 is set to specific functionality. The end-device belongs to a parent node which is either the coordinator or a router, and the parent node registers the end-device to its internal child table for message relaying. This table is maintained as long as the end-device polls the parent node periodically within the poll timeout value of the parent node. If the end-device does not poll for the poll timeout value of the parent node, the parent node removes the end-device from its child table and does not relay the message any more. Hence, the poll period of the end-device should be equal or shorter than the poll timeout of the parent node to avoid accidental removal of the end-device from the parent node’s child table.

The ZE20S poll interval can be configured by setting the S-registers using the AT commands as below.

ATS51=n, where n is poll timeout in seconds and the default is 60.
This value is for a parent node. If an end-device does not send a poll request to its parent coordinator or router within the poll timeout, the end-device is removed from the child table.

ATS56=n, where n is poll period in seconds and the default is 5.
End-device polls the parent node at poll period defined by S56 periodically. It is for the connection to the parent node and data transmission with other nodes in the network.

ATS55=n, where n is rejoin timeout and the default is 10.
When the connection to the parent node is broken for S55 in seconds, the child node tries to rejoin the current network.

Even though the end-device needs to poll the parent node continuously, it does not go into the sleep mode and it can send and receive messages any time unlike the sleepy end-device.

3.7 Setting up ZE20S as a Sleepy End-Device

A sleepy end-device can turn off radio to save power. To configure the ZE20S as a ZigBee sleepy end-device, please follow the steps below.

- Configure the CBKE Data Token
- Optionally, configure the Install Code
- Configure the ZE20S node as a sleepy end-device.
- Optionally, select the channel mask. Otherwise, the ZE20S will use the default channel mask.
- Optionally, set up the PAN ID and/or the extended PAN ID. Otherwise, the ZE20S will scan and join the ZigBee network found automatically.
- Configure the settings for sleep interval.

Please refer to the section ***3.5 Setting up ZE20S as a Router*** for the AT commands regarding the CBKE Data Token configuration, the Install Code configuration, the node type selection, the channel mask configuration and the PAN ID/Extended PAN ID configuration.

One of the unique features of the ZigBee device is its extreme low power consumption which is mainly achieved by its efficient sleep and wake-up mechanism. While ZigBee coordinator and routers are required to be in operation all the time, ZigBee sleepy end-devices can be configured to be in the sleep mode most of the time and only wake up at pre-configured time interval for data transmission before it goes into the sleep mode again. The ZE20S sleep interval can be configured by setting the S-registers using the AT commands as below.

ATS51=n, where n is poll timeout in seconds and the default is 60.

This value is for a parent node. If a sleepy end-device does not send a poll request to its parent coordinator or router within the poll timeout, the sleepy end-device is removed from the child table.

Unlike the non-sleepy end-device, sleepy end-device does not use S56 as the poll period. Instead, it uses S53 as the sleep duration which works similar to the poll period, except it goes to sleep mode.

ATS53=n, where n is sleep duration in quarter seconds and the default is 8.

The ZE20S sleeps for sleep duration and turns off radio.

ATS52=n, where n is poll period in milliseconds and the default is 100.

Sleepy end-device wakes up from the sleep mode if S53 timer expires or GPIO0 interrupt is generated or serial data is received. Once the sleepy end-device wakes up, it polls the parent node at poll period defined by S52 until it goes to sleep mode again. Since the sleepy end-device exchanges data with the parent node only by polling, S52 poll period is usually configured as around 100 ms.

ATS54=n, where n is standby timeout in seconds and the default is 5.

If there is no RF data or serial data for S54 seconds, the sleepy end-device goes to sleep mode again. If any serial or RF data is sent or received, the timer will reset and the sleepy end-device waits for another S54 seconds before it goes to sleep mode again.

ATS55=n, where n is rejoin timeout and the default is 10.

When the connection to the parent node is broken for S55 in seconds, the child node tries to rejoin the current network.

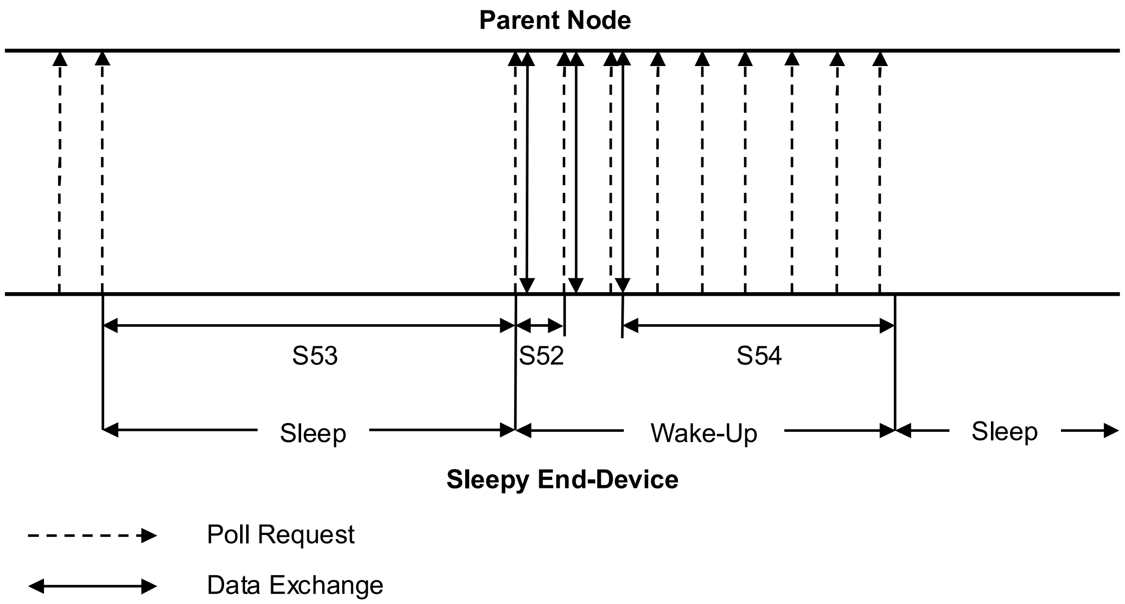


Figure 3-1 Sleep Interval and Wake-up Timing

The sleepy end-device can also wake up from the sleep mode by GPIO0 interrupt or incoming serial data. Once it wakes up, it also polls with the parent node at S52 interval and goes into sleep mode after S54 time elapses after the last data exchange.

Note: If the sleepy end-device wakes up by incoming serial data, the integrity of the first incoming byte is not guaranteed. It is strongly recommended to wake up the sleepy end-device by using GPIO0 first before sending the serial data or use higher level serial data protocol to check the data integrity.

3.8 Starting Key Establishment

To communicate with other node by Smart Energy cluster command, each node is required to exchange link keys and security data for authentication called “Key Establishment”. That process is driven by ‘Key Establish Cluster’ and is to be taken when the Smart Energy device joins to the Trust Center(coordinator). Key Establishment process is taken as follows:

- Coordinator forms a Smart Energy network.
- Router or end-device tries to join to the network and Trust Center(coordinator) requests the link key for joining process.
- After all joining process is completed, Trust Center or joining node send ‘Initiate CBKE Request’ command (Key Establishment Cluster) to the other.
- The other responses ‘CBKE Response’ and they exchange link key and ephemeral data.
- Each node regenerates link key to ‘Confirm key data’ and adds others ‘Confirm key data’ to Key

Table.

Key Establishment process can be triggered by running the AT commands as below.

AT+CBKESTART=addr,ep
,where addr is short or long address of destination node and ep is endpoint of destination node.

AT+CBKEPARTNER=addr,ep
,where addr is short or long address of destination node and ep is endpoint of destination node.

AT+CBKESTART is used to initiate Key Establishment between Trust Center(coordinator) and the other node(router or end-device). AT+CBKEPARTNER is used to exchange partner link key between non-Trust Center nodes(router or end-device).
Figure 3-2 shows how to use Key Establishment commands in each nodes.

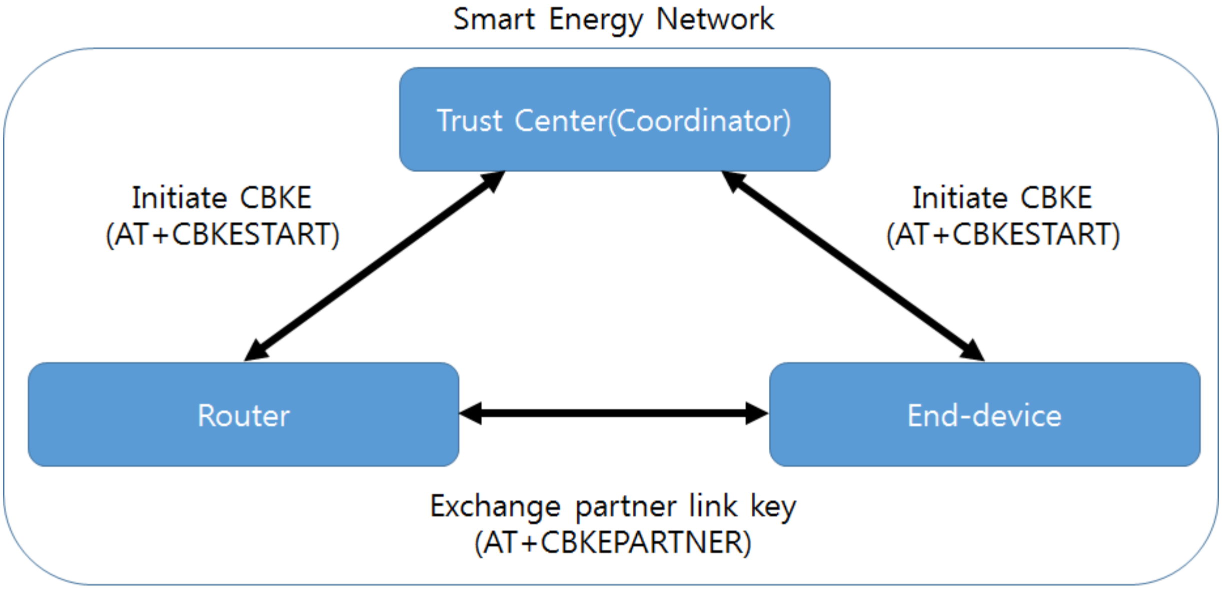


Figure 3-2 Key Establishment diagram

3.9 Frequency Agility

Frequency Agility refers to the function to shift operating channel to avoid interference. If frequency agility function of coordinator is enabled, the coordinator plays a role in network manager. It monitors communication failures, decides to changes the operating channel and send message to indicate the channel change. Other nodes in the network move to the designated channel by responding to the message.
The frequency agility needs to be set up only for the coordinator.

ATS71=n, where n is 0 = disabled, 1 = enabled.

4 Digital and Analog I/O

4.1 GPIO (General Purpose Inputs and Outputs) Configuration

The ZE20S module has total 18 I/O pins. Among these pins, I/O pin 2 is assigned to factory reset only and the user can use the rest 17 pins for general purpose inputs and outputs. The I/O pin assignments are shown in Table 4-1. As shown in Table 4-1, some pins provide special functions built into the firmware. Table 4-2 describes the special functions tied to the specific pins.

Table 4-1 Default I/O Pin Assignments

Pin Number	GPIO Number	Special Functions	AI Enable
2	-	Factory reset / Bootloader (**SW_0**)	X
3	0	Permit joining / Wake-up (**SW_1**)	X
4	1	UART_DTR	X
5	2	UART_DSR	X
6	3	UART_CTS	X
7	4	UART_RTS	X
8	5		X
9	6		X
10	7		X
11	8		X
32	9		O (AI_0)
31	10		O (AI_1)
30	11		O (AI_2)
29	12		O (AI_3)
28	13		O (AI_4)
27	14		O (AI_5)
24	15	Power LED (**LED_0**)	X
23	16	Status LED (**LED_1**)	X

The AT command below can be used to change the GPIO settings.

AT+GPIO=<values><CR> or AT+GPIO<number>=<value><CR>

, where <values> is a number between 0~6 as explained below:

0	*Disabled (unmonitored digital input)*
1	*Digital input, monitored*
2	*Digital output, default low*
3	*Digital output, default high*
4	*Analog input, single ended (only valid for GPIO9~14)*
5	*Reserved for pin-specific alternate functionalities (See Table 4-2)*

Smart Energy GPIO is treated in this document. Please refer to ProBee-ZE20S User Guide for usage of such options digital input, digital output and analog input.

Table 4-2 Special GPIO Pins

Function	GPIO#	Type	Description
Factory_reset / Bootloader	-	DI	This input is dedicated for the factory reset button. If user wants to restore the ZE20S device settings to factory default, this input (falling edge, i.e. High to Low change) needs to be pressed for more than two seconds. It will be applied after the input released again. If this input is pressed while the ZE20S starts, the local node enters bootloader menu and it is possible to upload the firmware. Factory reset button is not available when the ZE20S device is sleeping.
Permit_joining / Wake-up	0	DI	This input (falling edge, i.e. High to Low change) is used to permit joining or interrupt sleeping. This function is activated when GPIO0 is set to 5. The functions depend on the node type of the ZE20S. If the node is set to coordinator or router, this input works to permit other nodes joining for Permit joining timeout(S-register 22 value). If the node is set to sleepy end-device, it is used for wake-up signal when the node is sleeping.
UART_CTS	1	DI	This input is used for UART CTS if GPIO1 is set to 5. It should be enabled when CTS flow control is used.
UART_RTS	2	DO	This output is used for UART RTS if GPIO2 is set to 5. It should be enabled when RTS flow control is used.
UART_DTR	3	DO	This output is used for UART DTR if GPIO3 is set to 5.
UATR_DSR	4	DI	This input is used for UART DSR if GPIO4 is set to 5.
Power LED	15	DO	If GPIO15 is set to 5, this output turns ON (High) when power is supplied. If the node is set to an end-device, it will blink periodically. Please refer to ***8 S-Registers*** for S14 regarding the management of the LED status.
Status LED	16	DO	If GPIO16 is set to 5, this output turns ON (High) when the device joins a network, blinks when joining is permitted, and turns OFF (Low) when leaves the network. Please refer to ***8 S-Registers*** for S14 regarding the management of the LED status.

5 Working with Development Kit

The ZE20S development kit provides a quick and easy way to be familiar with the ZE20S and test its functionalities before you build your own PCB board to carry the ZE20S. It is always recommended to start with the ZE20S development kit before your own design.

5.1 Development Kit Contents

The development kit consists of ZE20S modules and development boards to carry them. The development board provides RS232 and USB interface to interface with computers and/or your target devices. It provides terminals for digital inputs/outputs and analog inputs, too. It also provides on/off switches and LEDs to simulate and verify digital inputs/outputs. In addition, it has temperature and light sensor so you can test more practical applications. Below is the list of the items included in the package.

- ProBee ZE20S ZigBee OEM Module w/ Chip Antenna (ProBee-ZE20SDC) (1)
- ProBee ZE20S ZigBee OEM Module w/ U.FL Connector (ProBee-ZE20SDU) (1)
- ProBee ZE20S ZigBee OEM Module w/ RPSMA Connector (ProBee-ZE20SDS) (1)
- 3 dBi Dipole Antenna, RP-SMA Plug, Right-hand Thread (2)
- 12cm U.FL to RPSMA Cable, Right-Hand Thread (1)
- Development Boards (3)
- DC Power Adapters (3)
- RS232 Serial Cable (3)
- USB Cable (3)

5.2 Development Board Features

The development board is designed to provide convenient access to the ZE20S interface pins. The development board provides external interfaces as below. Also Figure 5-1 shows the layout of those interfaces of the development board.

- RS232 Interface
- USB Interface
- 3 Pre-defined Digital Inputs: Hardware Reset, Factory Default Reset, Permit Joining
- 8 User Configurable Digital Input Buttons
- 8 Digital Outputs with LEDs (Active High)
- 4 Analog Inputs
- Temperature Sensor
- Light Sensor

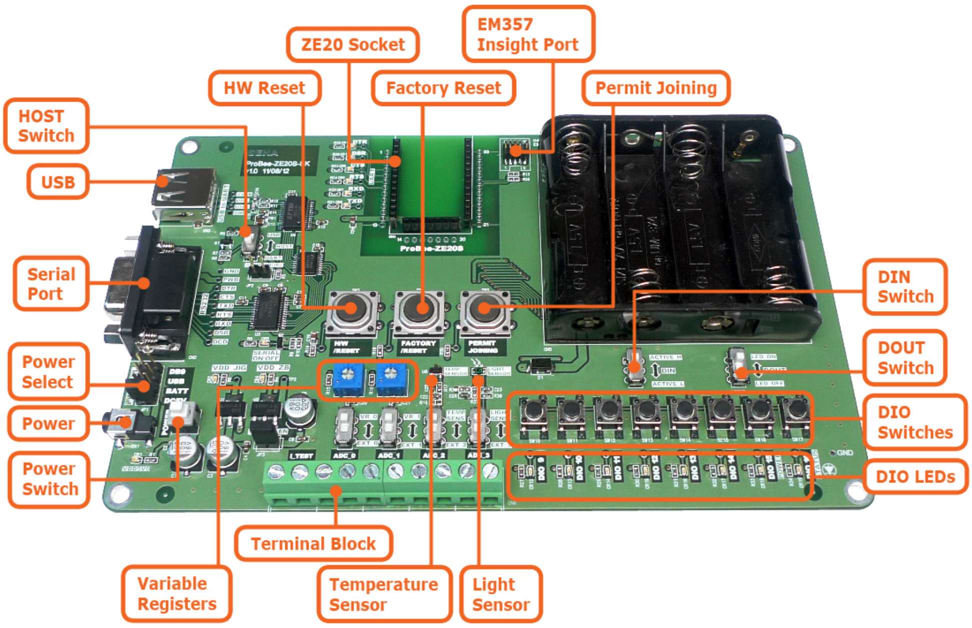


Figure 5-1 Development Board Layout

5.3 Development Board Hardware Descriptions

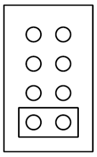
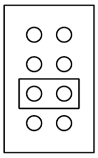
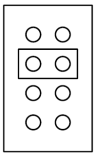
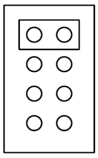
5.3.1 Power

There are four ways to supply power to the development board:

- 5V Power Adapter
- USB Port
- Batteries (AA 1.5V x 4)
- RS232 Port (5V through 9th Pin)

The PWR_SEL(JP1) jumper MUST be configured properly according to the way to supply power. Table 5-1 shows jumper configuration.

Table 5-1 Jumpers Configuration

		5V Power Adapter	Batteries	USB Port	RS232 Port
PWR_SEL	DB9				
	USB				
	BATT				
	DC5V				

5.3.2 RS232 Port

The development board has an RS232 interface for UART communication. The UART_TXD, UART_RXD, UART_DSR (GPIO_1), UART_DTR (GPIO_2), UART_CTS (GPIO_3) and UART_RTS (GPIO_4) pins of the ZE20S are connected with the RS232 port. To enable the RS232 port, the HOST (SW2) switch should be set to UART.

In addition to the data communication via RS232 port, the RS232 port is capable of supplying power to the development board. In order to use this feature, supply 5V power via the 9th pin of the DB9 socket.

5.3.3 USB Port

The development board also has a USB interface for UART communication. To use the USB port, the HOST (SW2) switch should be set to USB and the software driver should be installed on the host computer. The installation CD contains the software.

5.3.4 ZE20S GPIO Interface

The development board's external GPIO interfaces as well as functional buttons such as factory default reset switches are internally routed to the ZE20S GPIO pins. Table 5-2 summarizes the development board's interfaces and their corresponding GPIO pin numbers of the ZE20S.

Table 5-2 Development Board GPIO Interface Allocation

ZE20S Pin Number	Name	Description
2	Factory Reset	Factory Default Reset Switch (SW8)
3	GPIO_0	Permit Joining Switch (SW9)
4	GPIO_1	UART DTR Output
5	GPIO_2	UART DSR Input
6	GPIO_3	UART CTS Input
7	GPIO_4	UART RTS Output
22	H/W Reset	Hardware Reset Switch
23	GPIO_16	Digital Output with LED(CR19) Display (Active High) Digital Input with Button Switch (SW17, Active Low or Active High)
24	GPIO_15	Digital Output with LED(CR18) Display (Active High) Digital Input with Button Switch (SW16, Active Low or Active High)
25	UART_RXD	UART Data Input
26	UART_TXD	UART Data Output
27	GPIO_14	Digital Output with LED(CR17) Display (Active High) Digital Input with Button Switch (SW15, Active Low or Active High)
28	GPIO_13	Digital Output with LED(CR16) Display (Active High) Digital Input with Button Switch (SW14, Active Low or Active High)
29	GPIO_12	Digital Output with LED(CR15) Display (Active High) Digital Input with Button Switch (SW13, Active Low or Active High) Analog Input with Light Sensor (U7)
30	GPIO_11	Digital Output with LED(CR14) Display (Active High) Digital Input with Button Switch (SW12, Active Low or Active High) Analog Input with Temperature Sensor (U6)

31	GPIO_10	Digital Output with LED(CR13) Display (Active High) Digital Input with Button Switch (SW11, Active Low or Active High) Analog Input with Variable Register (VR2)
32	GPIO_9	Digital Output with LED(CR12) Display (Active High) Digital Input with Button Switch (SW10, Active Low or Active High) Analog Input with Variable Register (VR1)

5.3.5 LEDs (CR12-19)

The development board has 8 LEDs for digital output. To enable the LEDs, the GPIO_OUT (SW20) switch should be set to LED_ON. The LEDs are turned on when the related GPIO pins of the ZE20S are high.

Note: The ADC_0~3 switches (SW4~7) should be placed to EXT_0~3.

5.3.6 Button Switches (SW10-17)

The development board has 8 button switches for digital input. The switches can be configured as active high or active low with the GPIO_IN (SW19) switch.

Note: The ADC_0~3 switches (SW4~7) should be placed to EXT_0~3.
Note: The LED On/Off switch (SW20) should be placed to LED_OFF.
Note: If the GPIO_IN (SW19) is set to Active High, GPIO pull up/down option should be set to Pull-Down (ATS41=0). If the GPIO_IN (SW19) is set to Active Low, GPIO pull up/down option should be set to Pull-Up. (ATS41=1)

5.3.7 Variable Resistors (VR1, VR2)

There are two variable resistors for analog inputs on the development board. The analog inputs can be read with the GPIO_9 and GPIO_10. To enable the variable resistors, the ADC_0 and ADC_1 switches should be placed to VR_0 and VR_1.

Note: The LED On/Off switch (SW20) should be placed to LED_OFF.

5.3.8 Temperature Sensor

The GPIO_11 of the ZE20S is routed to the temperature sensor. The relationship between temperature and analog input is as follows:

Temperature(℃) ={ Volt(mV) -600(mV)} * 0.1(℃/mV)

Note: The LED On/Off switch (SW20) should be placed to LED_OFF.

5.3.9 Light Sensor

The GPIO_12 of the ZE20S is routed to the light sensor. Approximately, the relationship between illumination and analog input is as follows:

Illumination(Lux) = Volt(mV) * 0.25(Lux/mV)

Note: The LED On/Off switch (SW20) should be placed to LED_OFF.

5.3.10 Terminal Block

The development board has a terminal block for external analog inputs. Figure 5-2 shows the layout of the terminal block. To use external analog inputs instead of the variable resistors, temperature sensor or light sensor, the ADC_0 (SW4), ADC_1 (SW5), ADC_2 (SW6) or ADC_3 (SW7) should be placed on EXT_0, EXT_1, EXT_2, or EXT_3.

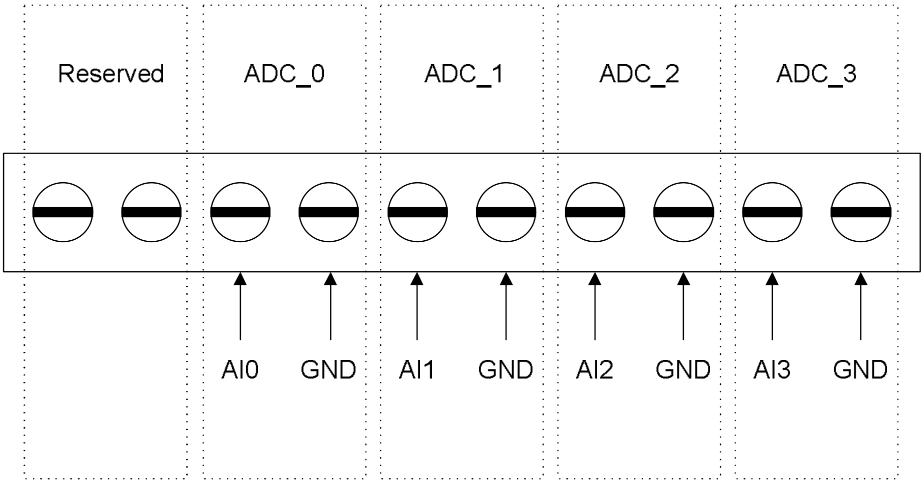


Figure 5-2 Terminal Block for External Anlalog Inputs

Note: The LED On/Off switch (SW20) should be placed to LED_OFF.

5.4 Configuration Example: 1 Coordinator and 2 End-Devices

Here is an example assuming different system configurations. This example can be described as follows:

- Setting CBKE Data Token and Install Code of all ZE20S modules for Smart Energy security.
- One ZE20S module acts as a ProBeeSESI coordinator and it is connected to a host computer via USB cable.
- One ZE20S module acts as a ProBeeSEID end device and it receives control commands from the others.
- One ZE20S module acts as a ProBeeSEME end device and it receives control commands from the others.

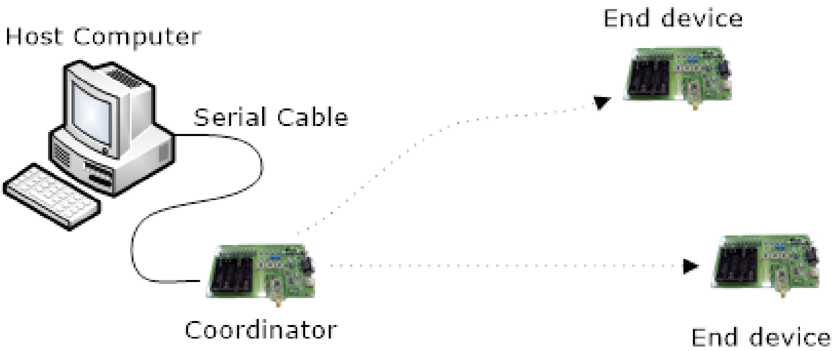


Figure 5-3 Example: 1 Coordinator and 2 End-Devices

5.4.1 CBKE Data Token Configuration

Hardware configuration for all devices is as follows:

- Set the HOST switch to UART
- Update ProBeeSESI firmware to a node for coordinator (See Chapter 2.3.3)
- Update ProBeeSEID firmware to a node for end-device (See Chapter 2.3.3)
- Update ProBeeSEME firmware to a node for end-device (See Chapter 2.3.3)

Each device should have CBKE Data Token before it acts as Smart Energy node for Smart Energy security. In this example, each device uses test certificate derived from its long address. These certificates in the example cannot be used for your devices. You should get certificate for your own devices. Please refer to the section ***3.2 Configuring CBKE Data Token*** for detail.

Test certificates for devices in this example are as follows:
Certificate#1 for ProBeeSESI(coordinator)
Long address: 0x000195aabbcc0000
CA Public Key: 0200fde8a7f3d1084224962a4e7c54e69ac3f04da6b8
Device Implicit Cert:
030505d6aed0c401c4003ae871fa2af4697fb89b5a8d000195aabbcc000054455354534543410109 1aabbaabbccddeef
Device Private Key: 00f156135beca22bc66388b26730e038525144be6c

Certificate#2 for ProBeeSEID(end-device)
Long address: 0x000195aabbcc0001
CA Public Key: 0200fde8a7f3d1084224962a4e7c54e69ac3f04da6b8
Device Implicit Cert:
0306a4e44deed93d6ce13ff81a6916e2c3473b1d0efe000195aabbcc00015445535453454341010 91aabbaabbccddeef
Device Private Key: 037ee06d90cadf5d399f0358260f70607e4f584f14

Certificate#3 for ProBeeSEME(end-device)
Long address: 0x000195aabbcc0002
CA Public Key:0200fde8a7f3d1084224962a4e7c54e69ac3f04da6b8
Device Implicit Cert:
20345824ccee7f6ca52724a7c1278207cfb7b5ea18b000195aabbcc0002544553545345434101091aabba abbccddeef
Device Private Key: 00e792620a20d2446ae14d90e7ed8c3d485201ffda

Example) Set test certificate to ProBeeSESI(coordinator)

```
#Screen of ProBeeSESI(coordinator)
AT+LONGADDR                   # Get long address
000195AABBCC0000              # Matched with Certificate#1
OK


AT+IMPLICIT                   # Show CBKE Data Token (same as AT+PUBLICKEY)
CBKE DATA TOKEN IS EMPTY      # Now CBKE Data Token is unlocked and empty
DEVICE IMPLICIT CERT(BUFFER):
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000
CA PUBLIC KEY & DEVICE PRIVATE KEY(BUFFER):
000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000
OK


AT+IMPLICIT=030505d6aed0c401c4003ae871fa2af4697fb89b5a8d000195aabbcc0000544553
545345434101091aabbaabbccddeef
                              # Input Device Implicit Cert
CBKE DATA TOKEN IS EMPTY      # Now CBKE Data Token is unlocked and empty
DEVICE IMPLICIT CERT(BUFFER):
030505D6AED0C401C4003AE871FA2AF4697FB89B5A8D000195AABBCC0000544553545345434101
091AABBAABBCCDDEEF
CA PUBLIC KEY & DEVICE PRIVATE KEY(BUFFER):
000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000
OK


AT+PUBLICKEY=0200fde8a7f3d1084224962a4e7c54e69ac3f04da6b800f156135beca22bc6638
8b26730e038525144be6c
                              # Input CA Public Key & Device Private Key
```

```
CBKE DATA TOKEN IS EMPTY      # Now CBKE Data Token is unlocked and empty
DEVICE IMPLICIT CERT(BUFFER):
030505D6AED0C401C4003AE871FA2AF4697FB89B5A8D000195AABBCC0000544553545345434101
091AABBAABBCCDDEEF
CA PUBLIC KEY & DEVICE PRIVATE KEY(BUFFER):
0200FDE8A7F3D1084224962A4E7C54E69AC3F04DA6B800F156135BECA22BC66388B26730E03852
5144BE6C
OK


AT+SETCBKEKEY                 # Lock CBKE Data Token
CBKE DATA TOKEN IS LOCKED     # Now CBKE Data Token is locked
DEVICE IMPLICIT CERT:
030505D6AED0C401C4003AE871FA2AF4697FB89B5A8D000195AABBCC0000544553545345434101
091AABBAABBCCDDEEF
CA PUBLIC KEY & DEVICE PRIVATE KEY:
0200FDE8A7F3D1084224962A4E7C54E69AC3F04DA6B800F156135BECA22BC66388B26730E03852
5144BE6C
OK
```

If CBKE Data Token is vaild and it is locked successfully, you can see 'GoodCert' message in AT+INFO.

```
AT+INFO
Security Level: 05, Security Profile: SE Full
SE Security Info: RealEcc, RealCbke, GoodCert   # GoodCert when
                                                # CBKE Data Token is
End-point count: 1                              # valid and locked
End-point No.1: ProfileID-0x0109, DeviceID-0x0500
Server-side cluster: 0x0000 (Basic)
Server-side cluster: 0x0003 (Identify)
Server-side cluster: 0x000A (Time)
Server-side cluster: 0x0701 (Demand Response and Load Control)
Server-side cluster: 0x0700 (Price)
Server-side cluster: 0x0703 (Messaging)
Client-side cluster: 0x0800 (Key establishment)
Server-side cluster: 0x0800 (Key establishment)
OK
```

Example) Set test certificate to ProBeeSEID(end-device)

```
#Screen of ProBeeSEID(end-device)
AT+LONGADDR                   # Get long address
```

```
000195AABBCC0001              # Matched with Certificate#1

OK


AT+IMPLICIT                   # Show CBKE Data Token (same as AT+PUBLICKEY)

CBKE DATA TOKEN IS EMPTY      # Now CBKE Data Token is unlocked and empty

DEVICE IMPLICIT CERT(BUFFER):
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000

CA PUBLIC KEY & DEVICE PRIVATE KEY(BUFFER):
000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000

OK


AT+IMPLICIT=0306a4e44deed93d6ce13ff81a6916e2c3473b1d0efe000195aabbcc0001544553
545345434101091aabbaabbccddeef

                              # Input Device Implicit Cert

CBKE DATA TOKEN IS EMPTY      # Now CBKE Data Token is unlocked and empty

DEVICE IMPLICIT CERT(BUFFER):
0306A4E44DEED93D6CE13FF81A6916E2C3473B1D0EFE000195AABBCC0001544553545345434101
091AABBAABBCCDDEEF

CA PUBLIC KEY & DEVICE PRIVATE KEY(BUFFER):
000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000

OK


AT+PUBLICKEY=0200fde8a7f3d1084224962a4e7c54e69ac3f04da6b8037ee06d90cadf5d399f0
358260f70607e4f584f14

                              # Input CA Public Key & Device Private Key

CBKE DATA TOKEN IS EMPTY      # Now CBKE Data Token is unlocked and empty

DEVICE IMPLICIT CERT(BUFFER):
0306A4E44DEED93D6CE13FF81A6916E2C3473B1D0EFE000195AABBCC0001544553545345434101
091AABBAABBCCDDEEF

CA PUBLIC KEY & DEVICE PRIVATE KEY(BUFFER):
0200FDE8A7F3D1084224962A4E7C54E69AC3F04DA6B8037EE06D90CADF5D399F0358260F70607E
4F584F14

OK


AT+SETCBKEKEY                 # Lock CBKE Data Token

CBKE DATA TOKEN IS LOCKED     # Now CBKE Data Token is locked

DEVICE IMPLICIT CERT(BUFFER):
0306A4E44DEED93D6CE13FF81A6916E2C3473B1D0EFE000195AABBCC0001544553545345434101
091AABBAABBCCDDEEF
```

```
CA PUBLIC KEY & DEVICE PRIVATE KEY(BUFFER):
0200FDE8A7F3D1084224962A4E7C54E69AC3F04DA6B8037EE06D90CADF5D399F0358260F70607E
4F584F14
OK
```

If CBKE Data Token is vaild and it is locked successfully, you can see ‘GoodCert’ message in AT+INFO.

```
AT+INFO
Security Level: 05, Security Profile: SE Full
SE Security Info: RealEcc, RealCbke, GoodCert   # GoodCert when
                                                # CBKE Data Token is
End-point count: 1                              # valid and locked
End-point No.1: ProfileID-0x0109, DeviceID-0x0502
Server-side cluster: 0x0000 (Basic)
Server-side cluster: 0x0003 (Identify)
Client-side cluster: 0x000A (Time)
Client-side cluster: 0x0701 (Demand Response and Load Control)
Client-side cluster: 0x0702 (Simple Metering)
Client-side cluster: 0x0700 (Price)
Client-side cluster: 0x0703 (Messaging)
Client-side cluster: 0x0800 (Key establishment)
Server-side cluster: 0x0800 (Key establishment)
OK
```

Example) Set test certificate to ProBeeSEME(end-device)

```
#Screen of ProBeeSEME(end-device)
AT+LONGADDR                  # Get long address
000195AABBCC0002             # Matched with Certificate#1
OK

AT+IMPLICIT                  # Show CBKE Data Token (same as AT+PUBLICKEY)
CBKE DATA TOKEN IS EMPTY     # Now CBKE Data Token is unlocked and empty
DEVICE IMPLICIT CERT(BUFFER):
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000
CA PUBLIC KEY & DEVICE PRIVATE KEY(BUFFER):
000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000
OK
```

```
AT+IMPLICIT=0306a4e44deed93d6ce13ff81a6916e2c3473b1d0efe000195aabbcc0001544553
545345434101091aabbaabbccddeef
                              # Input Device Implicit Cert
CBKE DATA TOKEN IS EMPTY      # Now CBKE Data Token is unlocked and empty
DEVICE IMPLICIT CERT(BUFFER):
0306A4E44DEED93D6CE13FF81A6916E2C3473B1D0EFE000195AABBCC0001544553545345434101
091AABBAABBCCDDEEF
CA PUBLIC KEY & DEVICE PRIVATE KEY(BUFFER):
000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000
OK


AT+PUBLICKEY=0200fde8a7f3d1084224962a4e7c54e69ac3f04da6b8037ee06d90cadf5d399f0
358260f70607e4f584f14
                              # Input CA Public Key & Device Private Key
CBKE DATA TOKEN IS EMPTY      # Now CBKE Data Token is unlocked and empty
DEVICE IMPLICIT CERT(BUFFER):
0306A4E44DEED93D6CE13FF81A6916E2C3473B1D0EFE000195AABBCC0001544553545345434101
091AABBAABBCCDDEEF
CA PUBLIC KEY & DEVICE PRIVATE KEY(BUFFER):
0200FDE8A7F3D1084224962A4E7C54E69AC3F04DA6B8037EE06D90CADF5D399F0358260F70607E
4F584F14
OK


AT+SETCBKEKEY                 # Lock CBKE Data Token
CBKE DATA TOKEN IS LOCKED     # Now CBKE Data Token is locked
DEVICE IMPLICIT CERT:
0306A4E44DEED93D6CE13FF81A6916E2C3473B1D0EFE000195AABBCC0001544553545345434101
091AABBAABBCCDDEEF
CA PUBLIC KEY & DEVICE PRIVATE KEY:
0200FDE8A7F3D1084224962A4E7C54E69AC3F04DA6B8037EE06D90CADF5D399F0358260F70607E
4F584F14
OK
```

If CBKE Data Token is vaild and it is locked successfully, you can see ‘GoodCert’ message in AT+INFO.

```
AT+INFO
Security Level: 05, Security Profile: SE Full
SE Security Info: RealEcc, RealCbke, GoodCert   # GoodCert when
                                                # CBKE Data Token is
End-point count: 1                              # valid and locked
```

```
End-point No.1: ProfileID-0x0109, DeviceID-0x0501
Server-side cluster: 0x0000 (Basic)
Client-side cluster: 0x000A (Time)
Server-side cluster: 0x0702 (Simple Metering)
Client-side cluster: 0x0703 (Messaging)
Client-side cluster: 0x0800 (Key establishment)
Server-side cluster: 0x0800 (Key establishment)
OK
```

5.4.2 Install Code Configuration

Each device can set Install Code optionally for enhanced Smart Energy security. Install Code is used to generate Pre-configured link key of joining node. Please refer to the section ***3.3 Setting up Install Code*** for detail.

Example) Set Install Code to ProBeeSEID(end-device)

```
#Screen of ProBeeSEID(end-device)
AT+INSTALLCODE                                  # Get current Install Code
INSTALL CODE: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF  # Now empty
OK

AT+INSTALLCODE=1122334455667788                 # Set Install Code
OK                                              # Install code must be
                                                # 6/8/12/16 byte hexadecimal
AT+INSTALLCODE
INSTALL CODE: 1122334455667788                  # Successfully changed
OK
```

Example) Set Install Code to ProBeeSEME(end-device)

```
#Screen of ProBeeSEME(end-device)
AT+INSTALLCODE                                  # Get current Install Code
INSTALL CODE: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF  # Now empty
OK

AT+INSTALLCODE=1122334455667788                 # Set Install Code
OK                                              # Install code must be
                                                # 6/8/12/16 byte hexadecimal
AT+INSTALLCODE
```

```
INSTALL CODE: 1122334455667788                  # Successfully changed
OK
```

5.4.3 ProBeeSESI Coordinator Configuration

Hardware configuration for the coordinator is as follows:

- Set the HOST switch to UART
- Connect a serial cable between a host computer and the coordinator

The AT commands for coordinator configuration is as follows:

```
#Screen of ProBeeSESI(coordinator)
AT+NODETYPE=1
OK
AT+PANID=7772
OK
ATZ
OK
Channel:14, PanID:7772                    # Forming network successfully
```

5.4.4 ProBeeSEID End-device Configuration

Hardware configuration for the end-device is as follows:

- Set the HOST switch to UART
- Connect a serial cable between a host computer and the end-device

The AT commands for the ProBeeSEID end-device are as follows:

```
#Screen of ProBeeSEID(end-device)
AT+NODETYPE=3
OK
AT+PANID=7772
OK
ATZ
OK
Channel:14, PanID:7772
FAILED TO JOIN(AD)                  # Failed to join with security error(AD)

AT+LONGADDR                         # Get long address
000195AABBCC0001
OK
```

```
AT+KEYTABLE                         # Get Pre-configured Link Key
NETWORK KEY: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

LINK KEY TABLE
#|IEEE_ADDRESS   |TYPE|AUTH|KEY
*|0000000000000000|LINK|NO  |41618FC0C83B0E14A589954B16E31466
1/6 ENTRIES USED
OK
```

'FAILED TO JOIN(AD)' means that ProBeeSEID node failed to join the network because Trust Center(coordinator) had no Pre-configured link key of ProBeeSEID and denied to join. To fix it, Trust Center(coordinator) node must add Pre-configured link key of end-device.

The AT commands for setting Pre-configured link key for Key Table of Trust Center(coordinator) is as follows:

```
#Screen of ProBeeSESI(coordinator)
AT+KEYTABLE=0,000195AABBCC0001,41618FC0C83B0E14A589954B16E31466
OK
            # Add Pre-configured Link Key of end-device(000195AABBCC0001)
            # to No.0 Key Table entry in coordinator

AT+KEYTABLE
NETWORK KEY: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

LINK KEY TABLE
#|IEEE_ADDRESS   |TYPE|AUTH|KEY
0|000195AABBCC0001|LINK|NO  |41618FC0C83B0E14A589954B16E31466
1/6 ENTRIES USED
OK
```

Then, you can join successfully.

```
#Screen of ProBeeSEID(end-device)
Channel:14, PanID:7772              # Do not put any error message
                                    # Successfully joined

AT+KEYTABLE
```

```
NETWORK KEY: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

LINK KEY TABLE
#|IEEE_ADDRESS    |TYPE|AUTH|KEY
*|000195AABBCC0000|LINK|NO  |41618FC0C83B0E14A589954B16E31466
1/6 ENTRIES USED
OK                                  # IEEE_ADDRESS changed to
                                    # long address of Trust Center
```

5.4.5 ProBeeSEME End-device Configuration

Hardware configuration for the end-device is as follows:

- Set the HOST switch to UART
- Connect a serial cable between a host computer and the end-device

The AT commands for the ProBeeSEME end-device are as follows:
(Same as ProBeeSEID end-device)

```
#Screen of ProBeeSEME(end-device)
AT+NODETYPE=3
OK
AT+PANID=7772
OK
AT+LONGADDR
000195AABBCC0002
OK
ATZ
OK
Channel:14, PanID:7772
FAILED TO JOIN(AD)                  # Failed to join with security error(AD)

AT+KEYTABLE                         # Get Pre-configured Link Key
NETWORK KEY: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

LINK KEY TABLE
#|IEEE_ADDRESS    |TYPE|AUTH|KEY
*|0000000000000000|LINK|NO  |41618FC0C83B0E14A589954B16E31466
1/6 ENTRIES USED
OK
```

'FAILED TO JOIN(AD)' means that ProBeeSEME node failed to join the network because Trust Center(coordinator) had no Pre-configured link key of ProBeeSEME and denied to join. To fix it, Trust Center(coordinator) node must add Pre-configured link key of end-device.

The AT commands for setting Pre-configured link key for Key Table of Trust Center(coordinator) is as follows:

```
#Screen of ProBeeSESI(coordinator)
AT+KEYTABLE=1,000195AABBCC0002,41618FC0C83B0E14A589954B16E31466
OK
            # Add Pre-configured Link Key of end-device(0001950000000003)
            # to No.1 Key Table entry in coordinator
AT+KEYTABLE
NETWORK KEY: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

LINK KEY TABLE
#|IEEE_ADDRESS    |TYPE|AUTH|KEY
0|000195AABBCC0001|LINK|NO  |41618FC0C83B0E14A589954B16E31466
1|000195AABBCC0002|LINK|NO  |41618FC0C83B0E14A589954B16E31466
2/6 ENTRIES USED
OK
```

Then, you can join successfully.

```
#Screen of ProBeeSEME(end-device)
Channel:14, PanID:7772              # Do not put any error message
                                    # Successfully joined
AT+KEYTABLE
NETWORK KEY: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

LINK KEY TABLE
#|IEEE_ADDRESS    |TYPE|AUTH|KEY
*|000195AABBCC0000|LINK|NO  |41618FC0C83B0E14A589954B16E31466
1/6 ENTRIES USED
OK                                  # IEEE_ADDRESS changed to
                                    # long address of Trust Center
```

5.4.6 Key Establishment Process

To complete security authentication, use AT+CBKESTART and AT+CBKEPARTNER command.

- Use AT+CBKESTART for Trust Center(coordinator)↔end-device Key Establishment.
- Use AT+CBKEPARTNER for Partner Key Exchange between nodes other than Trust Center.

Example) Initiate Key Establishment from ProBeeSEID(end-device) to ProBeeSESI(coordinator)

```
#Screen of ProBeeSEID(end-device)
AT+DEST=0000,01               # Set destination node to coordinator(0x0000),
                              # End-point is 0x01
DESTINATION ADDRESS: 0x0000, END POINT: 0x01
OK
AT+CBKESTART
CBKE DEST: 0x0000, EP: 0x01, RESULT: 0x00
OK
+0000|SE_INIT_CBKE_RESPONSE
+0000|SE_EPHEMERAL_DATA_RESPONSE
+0000|SE_CONFIRM_KEY_DATA_RESPONSE
```

Then ProBeeSESI(coordinator) node receives following messages:

```
#Screen of ProBeeSESI(coordinator)
+CEDF|SE_INIT_CBKE_REQUEST          # +CEDF is short address of end-device,
+CEDF|SE_EPHEMERAL_DATA_REQUEST     # it can be shown differently
+CEDF|SE_CONFIRM_KEY_DATA_REQUEST   # in your screen
+CEDF|SE_GLOBAL_COMMAND|0800|0B
```

Key Table of ProBeeSESI is changed as follows:

```
#Screen of ProBeeSESI(coordinator)
AT+KEYTABLE
NETWORK KEY: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

LINK KEY TABLE
#|IEEE_ADDRESS    |TYPE|AUTH|KEY
0|000195AABBCC0001|LINK|YES |C9AF9B248454840FDD80AAFBBAA4028F
1|000195AABBCC0002|LINK|NO  |41618FC0C83B0E14A589954B16E31466
                  # Key Table No.0 is changed
                  # AUTH(Authentication) is changed NO to YES
                  # KEY(Link Key) is changed by initiate Key Establishment
2/6 ENTRIES USED
OK
```

Key Table of ProBeeSEID is also changed as follows:

```
#Screen of ProBeeSEID(end-device)
AT+KEYTABLE
NETWORK KEY: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

LINK KEY TABLE
#|IEEE_ADDRESS    |TYPE|AUTH|KEY
*|000195AABBCC0000|LINK|YES |C9AF9B248454840FDD80AAFBBAA4028F
1/6 ENTRIES USED
                  # Trust Center Link Key(Number *) is changed
                  # AUTH(Authentication) is changed NO to YES
                  # KEY(Link Key) is changed by initiate Key Establishment
1/6 ENTRIES USED
OK
```

Example) Initiate Key Establishment from ProBeeSESI(coordinator) to ProBeeSEME(end-device)

```
#Screen of ProBeeSESI(coordinator)
AT+DEST=000195AABBCC0002,01   # Set destination node to ProBeeSEME(0xAB74),
                              # End-point is 0x01
DESTINATION ADDRESS: 0xAB74, END POINT: 0x01
OK
AT+CBKESTART
CBKE DEST: 0xAB74, EP: 0x01, RESULT: 0x00
OK

+AB74|SE_INIT_CBKE_RESPONSE         # +AB74 is short address of end-device,
+AB74|SE_EPHEMERAL_DATA_RESPONSE    # it can be shown differently
+AB74|SE_CONFIRM_KEY_DATA_RESPONSE  # in your screen
```

Then ProBeeSEME(end-device) node receives following messages:

```
#Screen of ProBeeSEME(end-device)
+0000|SE_INIT_CBKE_REQUEST
```

```
+0000|SE_EPHEMERAL_DATA_REQUEST
+0000|SE_CONFIRM_KEY_DATA_REQUEST
+0000|SE_GLOBAL_COMMAND|0800|0B
```

Key Table of ProBeeSEME is changed as follows:

```
#Screen of ProBeeSEME(end-device)
AT+KEYTABLE
NETWORK KEY: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

LINK KEY TABLE
#|IEEE_ADDRESS    |TYPE|AUTH|KEY
*|000195AABBCC0000|LINK|YES |5C500189361BC97EB1F50AE429B98565
                  # AUTH(Authentication)  is changed NO to YES
                  # KEY(Link Key) is changed by initiate Key Establishment
                  # IEEE_ADDRESS is changed to Trust Center Address

1/6 ENTRIES USED  # ProBeeSEME do not have any Link Key in Key Table
                  # Just have Trust Center Link Key
```

Key Table of ProBeeSESI is also changed as follows:

```
#Screen of ProBeeSESI(coordinator)
AT+KEYTABLE
NETWORK KEY: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

LINK KEY TABLE
#|IEEE_ADDRESS    |TYPE|AUTH|KEY
0|000195AABBCC0001|LINK|YES |C9AF9B248454840FDD80AAFBBAA4028F
1|000195AABBCC0002|LINK|YES |5C500189361BC97EB1F50AE429B98565
                  # Key Table No.1 is changed
                  # AUTH(Authentication) is changed NO to YES
                  # KEY(Link Key) is changed by initiate Key Establishment
2/6 ENTRIES USED
OK
```

Example3) Set Partner Key Exchange from ProBeeSEID(end-device) to ProBeeSEME(end-device)

```
#Screen of ProBeeSEID(end-device)
AT+CBKEPARTNER=AB74,01        # You can also use AT+DEST for setting
                              # destination node
CBKE PARTNER DEST: 0xAB74, EP: 0x01, RESULT: 0x00
OK
```

Key Table of ProBeeSEID is changed as follows:

```
#Screen of ProBeeSEID(end-device)
AT+KEYTABLE
NETWORK KEY: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

LINK KEY TABLE
#|IEEE_ADDRESS    |TYPE|AUTH|KEY
*|000195AABBCC0000|LINK|YES |C9AF9B248454840FDD80AAFBBAA4028F
0|000195AABBCC0002|LINK|YES |D0B64ED5A1D33F4CD28EE3BAE6B9E3A7
                  # Key Table No.0 is added
                  # Partner key exchanged successfully
2/6 ENTRIES USED
OK
```

Key Table of ProBeeSEME is also changed as follows:

```
#Screen of ProBeeSEME(end-device)
AT+KEYTABLE
NETWORK KEY: FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

LINK KEY TABLE
#|IEEE_ADDRESS    |TYPE|AUTH|KEY
*|000195AABBCC0000|LINK|YES |5C500189361BC97EB1F50AE429B98565
0|000195AABBCC0001|LINK|YES |D0B64ED5A1D33F4CD28EE3BAE6B9E3A7
                  # Key Table No.0 is added
                  # Partner key exchanged successfully
2/6 ENTRIES USED
OK
```

5.4.7 Cluster Command for Binding

Each Smart Energy node can communicate with the other by cluster command. A cluster consists of server-side cluster and client-side cluster, and both of them have their own commands. If a Smart Energy node has server-side or client-side cluster, the node can send cluster command to other node that has the other side cluster. Here is an example for sending cluster command.

Example) Send the ‘Get Last Message’ command in Messaging cluster from ProBeeSEID node.

```
#Screen of ProBeeSEID(end-device)
AT+DEST=0000,01<CR>           # Set Destination node to coordinator
                              # coordinator is now ProBeeSESI
AT+MSGETLAST<CR>              # Send ‘Get Last Message’ Command to
                              # ProBeeSESI Messaging Cluster (Server)
OK                            # Response
```

Then ProBeeSESI node receives the ‘Get Last Message’ command messages as below:

```
#Screen of ProBeeSESI(coordinator)
+CEDF|SE_GET_LAST_MESSAGE     # Received ‘Get Last Message’ Command from
                              # ProBeeSEID
```

Example) Send the ‘Display Message’ command in Messaging cluster from ProBeeSESI node.

```
#Screen of ProBeeSESI(coordinator)
AT+DEST=AB74,01<CR>           # Set Destination node to ProBeeSEME
                              # ProBeeSEME is now end-device
AT+MSID=11335577<CR>          # Set Message ID
OK
AT+MSSTARTTIME=08-22-2013/15:08:30<CR>
                              # Set Messaging Start Time
OK
AT+MSCTRL=01<CR>              # Set Message Control
OK
AT+MSDURATION=0200<CR>        # Set Message Duration
OK
AT+MSDISPLAY=Test Message     # Send ‘Display Message’ command:
                              # “Test Message” string to destination node
Test Message
OK                            # Response
```

Then ProBeeSEME node receives following message:

```
#Screen of ProBeeSEME(end-device)
+0000|SE_DISPLAY_MESSAGE|11335577|01|0200|Test Message|08-22-2013/15:08:30
```

5.4.8 Showing Attribute Information

Each Smart Energy node has its own attributes. User can see their information by running the AT command as below:

Example) Show attributes of ProBeeSESI node

```
#Screen of ProBeeSESI(coordinator)
AT+ATTR<CR>                         # Show attributes of current device
clus|attr|data(clusterName)
0000|0000|01 (Basic)
0000|0001|00 (Basic)
0000|0002|00 (Basic)
0000|0003|00 (Basic)
0000|0007|00 (Basic)
0000|0011|00 (Basic)
0000|0012|01 (Basic)
0000|0013|00 (Basic)
0000|0014|00 (Basic)
0003|0000|0000 (Identify)
0701|0000|client-side (Demand Response and Load Control)
0701|0001|client-side (Demand Response and Load Control)
0701|0002|client-side (Demand Response and Load Control)
0701|0003|client-side (Demand Response and Load Control)
0800|0000|client-side (Key establishment)
0800|0000|0000 (Key establishment)
OK                                  # Response
                                    # clus: cluster id
                                    # attr: attribute id
                                    # data: stored data
```

5.4.9 Reading Attribute

User can read attribute of local node or remote node by running the AT command as below:

- AT+SREAD can be used for reading attribute of local node.
- AT+CREAD can be used for reading attribute of remote node.
- User should set destination address by AT+DEST before running AT+CREAD.
- Destination node must implement corresponding attribute and cluster.

Example) Read attribute of local node.

```
#Screen of ProBeeSESI(coordinator)
AT+SREAD=0000,0007<CR>        # Read device attribute
                              # Basic cluster(0x0000),
                              # Power source attribute(0x0007)
2                             # Unlocked
OK                            # Response
```

Example) Read attribute of remote node.

```
#Screen of ProBeeSEID(end-device)
AT+DEST=0000,01<CR>           # Set destination node address
OK                            # Response
AT+CREAD=0702,0000<CR>        # Read attribute of destination node
                              # Metering cluster(0x0702),
                              # Current Summation delivered attribute(0x0000)
00 00 00 00 00 00             # 5-byte value
OK                            # Response
```

5.4.10 Writing Attribute

User can write attribute to local node or remote node by running the AT command as below:

- AT+SWRITE can be used for writing attribute to local node.
- AT+CWRITE can be used for writing attribute to remote node.
- User should set destination address by AT+DEST before running AT+CWRITE.
- Destination node must implement corresponding attribute and cluster.

Example) Write attribute to local node.

```
#Screen of ProBeeSESI(coordinator)
AT+SWRITE=0000,0012,00<CR>    # Write device attribute
                              # Basic cluster(0x0000),
                              # Device Enabled attribute(0x0012)
                              # Set to 0x00 (Disabled)
OK                            # Response
```

Example) Write attribute to remote node.

```
#Screen of ProBeeSEME(coordinator)
AT+DEST=0001950000000003,01<CR>     # Set destination node address
OK                                  # Response
AT+CWRITE=0702,0001,010203040506<CR># Write to attribute of destination node
                                    # Metering cluster(0x0702),
                                    # Current Max Demand Delivered
                                    # attribute(0x0002)
                                    # Set to 0x010203040506
```

5.4.11 Binding Metering Cluster

ProBeeSEME node has server-side Metering Cluster and ProBeeSEID has client-side Metering Cluster. They can communicate with each other by Metering Cluster commands.

Example) Set ‘Metering Profile’ and ‘Metering End Time’ at ProBeeSEME node.

```
#Screen of ProBeeSEME(end-device)
AT+SMSETPR=12,AABBCC,010203<CR>           # Set Metering Profile
OK
AT+SMSETPR<CR>
ProfileIntervalPeriod: 12, NumberOfPeriodsDelivered: 2
Intervals:
AA BB CC
01 02 03
OK                                        # Response

AT+SMENDTIME=08-04-2013/12:34:56<CR>      # Set Metering End Time
OK
```

Example) Send ‘Get Profile’ command from ProBeeSEID to ProBeeSEME with ‘Delivered’.

```
#Screen of ProBeeSEID(end-device)
AT+DEST=AB74,01<CR>                 # Set destination address to ProBeeSEME
DESTINATION ADDRESS: 0xAB74, END POINT: 0x01
OK
AT+SMGETPROFD=01<CR>                # Send 'Get Profile' command
OK
```

Then ProBeeSEID node receives the ‘Get Profile Response’ command messages as below:

```
+AB74|SE_GET_PROFILE_RESPONSE|00|12|02|08-04-2013/12:34:56
+AB74|SE_GET_PROFILE_RESPONSE|INTERVAL0|AABBCC
+AB74|SE_GET_PROFILE_RESPONSE|INTERVAL1|010203
+AB74|SE_GLOBAL_COMMAND|0702|0B     # Receive response message from ProBeeSEME
```

5.4.12 Binding DRLC Cluster

ProBeeSESI node has server-side DRLC Cluster and ProBeeSEID has client-side DRLC Cluster. They can communicate with each other by DRLC Cluster commands.

Example) Add, Modify, Remove ‘DRLC Event List’ at ProBeeSESI node.

```
#Screen of ProBeeSESI(coordinator)
AT+DLSTARTTIME=08-04-2013/13:14:15<CR>    # Set DRLC Start Time
OK
AT+DLSET=00000001,000A,01,0100,02,30,1000,40,1200,0A,0B,0C<CR>
OK                                        # Add DRLC Event (ID: 00000001)
AT+DLSET=ABAB,35,02,200,05,15,500,25,600,10,3,4<CR>
OK                                        # Add DRLC Event (ID: 0000ABAB)
AT+DLLIST<CR>                             # Show DRLC Event List
event ID|clss|gr|dura|cr|cool(sp)|heat(sp)|av|du|ct|start time
00000001|000A|01|0100|02|30(1000)|40(1200)|0A|0B|0C|08-04-2013/13:14:15
0000ABAB|0035|02|0200|05|15(0500)|25(0600)|10|03|04|08-04-2013/13:14:15
OK

AT+DLSET=00000001,00BB,05,0200,55,40,1000,50,2000,01,02,03<CR>
OK                                        # Modify DRLC Event by inputting
                                          # same ID in DRLC List
AT+DLLIST                                 # Show DRLC Event List
```

```
event ID|clss|gr|dura|cr|cool(sp)|heat(sp)|av|du|ct|start time
00000001|00BB|05|0200|55|40(1000)|50(2000)|01|02|03|08-04-2013/13:14:15
0000ABAB|0035|02|0200|05|15(0500)|25(0600)|10|03|04|08-04-2013/13:14:15
OK                                        # Event ID 00000001 is modified
AT+DLDEL=00000001<CR>                     # Remove DRLC Event
OK                                        # Response
AT+DLLIST<CR>                             # Show DRLC Event List
event ID|clss|gr|dura|cr|cool(sp)|heat(sp)|av|du|ct|start time
0000ABAB|0035|02|0200|05|15(0500)|25(0600)|10|03|04|08-04-2013/13:14:15
OK
```

Client-side DRLC Cluster(Included in ProBeeSEID) can get DRLC event from server-side DRLC Cluster(Included in ProBeeSESI) by sending ‘Get Scheduled Events’ command.

```
#Screen of ProBeeSEID(end-device)
AT+DEST=0000,01<CR>                 # Set destination address to ProBeeSESI
DESTINATION ADDRESS: 0x0000, END POINT: 0x01
OK
AT+DLGETSCH<CR>                     # Send ‘Get Scheduled Events’ command
OK
+0000|SE_GLOBAL_COMMAND|0701|0B
+0000|SE_LOAD_CONTROL_EVENTS|0000ABAB
                                    # Receive response message from ProBeeSESI
AT+DLLIST<CR>                       # Show DRLC Event List
event ID|clss|gr|dura|cr|cool(sp)|heat(sp)|av|du|ct|start time
0000ABAB|0035|02|0200|05|15(0500)|25(0600)|10|03|04|08-04-2013/13:14:15
OK
```

Also server-side DRLC Cluster(Included in ProBeeSESI) can send or cancel DRLC Event to client-side DRLC Cluster(Included in ProBeeSEID).

```
#Screen of ProBeeSESI(coordinator)
AT+DEST=CEDF,01<CR>     # Set destination address to ProBeeSEID
DESTINATION ADDRESS: 0xCEDF, END POINT: 0x01
OK

AT+DLLOADCTRL=0000ABAB<CR>
OK                      # Load DRLC Event (ID 0000ABAB) to ProBeeSEID
                        # Now Event ID 0000ABAB is loaded on ProBeeSEID
```

```
AT+DLCANCELLOAD=0000ABAB<CR>
OK                      # Cancel DRLC Event (ID 0000ABAB) to ProBeeSEID
                        # Now Event ID 0000ABAB is canceled on ProBeeSEID
```

Then ProBeeSEID(end-device) receives following messages:

```
#Screen of ProBeeSEID(end-device)
+0000|SE_LOAD_CONTROL_EVENTS|0000ABAB
                  # Receive Load Control Events command from ProBeeSESI
AT+DLLIST<CR>     # Show DRLC Event List
event ID|clss|gr|dura|cr|cool(sp)|heat(sp)|av|du|ct|start time
0000ABAB|0035|02|0200|05|15(0500)|25(0600)|10|03|04|08-04-2013/13:14:15
OK

+0000|SE_CANCEL_LOAD_CONTROL_EVENTS|0000ABAB|00000000|00000000
                  # Receive Cancel Load Control Events command from ProBeeSESI
AT+DLLIST<CR>     # Show DRLC Event List
event ID|clss|gr|dura|cr|cool(sp)|heat(sp)|av|du|ct|start time
OK
```

5.4.13 Binding Price Cluster

ProBeeSESI node has server-side Price Cluster and ProBeeSEID has client-side Price Cluster. They can communicate with each other by Price Cluster commands.

Example) Send ‘Publish Price’ command from ProBeeSESI node to ProBeeSEID node.

```
#Screen of ProBeeSESI(coordinator)
AT+DEST=CEDF,01<CR>     # Set destination address to ProBeeSEID
DESTINATION ADDRESS: 0xCEDF, END POINT: 0x01
OK
AT+PRCURRENTTIME=08-04-2013/14:20:30<CR>
OK
AT+PRSTARTTIME=08-04-2013/15:20:30<CR>
OK
AT+PRSETPUBLISH=0000EEEE,0000CCCC,24,1122,01,03,0200,11223344,55<CR>
                              # Send ‘Publish Price’ command to ProBeeSEID
OK
+CEDF|SE_GLOBAL_COMMAND|0700|0B     # Response
```

Then ProBeeSEID node receives following message:

```
#Screen of ProBeeSEID(end-device)

+0000|SE_PUBLISH_PRICE|0000EEEE|SAMPLE|0000CCCC|24|1122|01|03|0200|08-04-
2013/14:20:30|11223344|55|00000000|00|00000000|00|00|00|00|08-04-2013/15:20:30
```

6 SE(Smart Energy) Profile Descriptions

6.1 Device Descriptions

Device descriptions specified in this profile are summarized in following Table 6-1, “Devices Specified in the SE Profile”. The devices are organized according the end application areas they address. A product that conforms to this specification shall implement at least one of these device descriptions and shall also include the device descriptions corresponding to all applications implemented on the product where a standard device description is specified in this profile. For example, if a product implements both a light dimmer and a light sensor application, then the Dimmable Light and Light Sensor device descriptions must both be supported. This list will be added to in future versions of the profile as new clusters are developed to meet the needs of manufacturers. The reserved values shall not be used until the profile defines them. Manufacturer-specific device descriptions shall reside on a separate endpoint and use a private profile ID.

Table 6-1 Devices Specified in the SE Profile

	Device	Device ID
Generic	Range Extender	0x0008
Smart Energy	***Energy Service Interface (Supported in ProBeeSESI)***	***0x0500***
	Metering Device (Supported in ProBeeSEME)	***0x0501***
	In-Premises Display (Supported in ProBeeSEID)	***0x0502***
	Programmable Communicating Thermostat (PCT) Load Control Device	0x0503 0x0504
	Smart Appliance	0x0505
	Prepayment Terminal	0x0506

6.2 ZigBee Cluster Library (ZCL)

This profile utilizes the clusters specified in the ZigBee Cluster Library. The implementation details for each cluster are given in the ZCL specifications. Further specification and clarification are given in this profile where necessary. The ZCL provides a mechanism for clusters to report changes to the value of various attributes. It also provides commands to configure the reporting parameters. The attributes that a particular cluster is capable of reporting are listed in the ZCL specification for each cluster. Products shall support the reporting mechanism for all attributes specified in the ZCL that the product implements within a given cluster. The tables in ***6.4 ProBee SE Profiles Description*** show which clusters are supported in ProBeeSExx firmwares and which attributes / commands are available in each cluster.

6.3 Cluster Binding

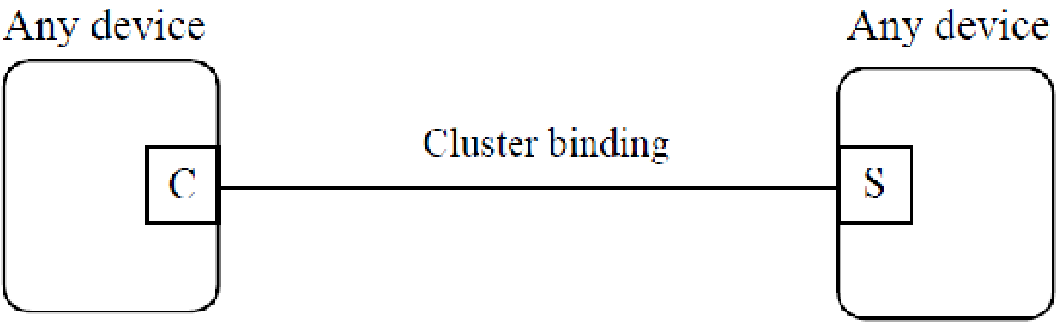


Figure 6-1 The ZCL Client Server Model

A cluster is a related collection of commands and attributes, which together define an interface to specific functionality. Typically, the entity that stores the attributes of a cluster is referred to as the server of that cluster and an entity that affects or manipulates those attributes is referred to as the client of that cluster. However, if required, attributes may also be present on the client of a cluster.

Commands that allow devices to manipulate attributes, the “read attribute” (AT+CREAD, AT+SREAD) or “write attribute” (AT+CWRITE, AT+SWRITE) commands, are sent from a client device and received by the server device. Any response to those commands, the “read attribute” response or the “write attribute” response commands, are sent from the server device and received by the client device.

The clusters supported by an application object within an application profile are identified through the simple descriptor (see Chapter 3), specified on each active endpoint of a device. In the simple descriptor, the application input cluster list shall contain the list of server clusters supported on the device and the application output cluster list shall contain the list of client clusters supported on the device.

ProBee supports 3 SE profiles (ProBeeSESI, ProBeeSEID, and ProBeeSEME) and at least two nodes with SE profile are necessary to test cluster binding and SE interaction. Following figure shows which profile clusters can communicate other ProBee SE profiles.

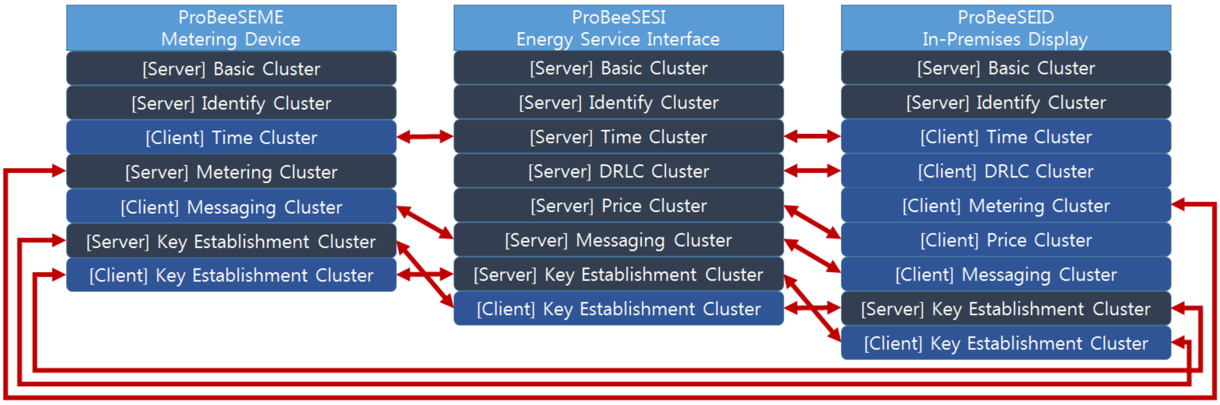

ProBeeSEME
Metering Device
[Server] Basic Cluster
[Server] Identify Cluster
[Client] Time Cluster
[Server] Metering Cluster
[Client] Messaging Cluster
[Server] Key Establishment Cluster
[Client] Key Establishment Cluster
ProBeeSESI
Energy Service Interface
[Server] Basic Cluster
[Server] Identify Cluster
[Server] Time Cluster
[Server] DRLC Cluster
[Server] Price Cluster
[Server] Messaging Cluster
[Server] Key Establishment Cluster
[Client] Key Establishment Cluster
ProBeeSEID
In-Premises Display
[Server] Basic Cluster
[Server] Identify Cluster
[Client] Time Cluster
[Client] DRLC Cluster
[Client] Metering Cluster
[Client] Price Cluster
[Client] Messaging Cluster
[Server] Key Establishment Cluster
[Client] Key Establishment Cluster

Figure 6-2 ProBee SE profile cluster binding diagram

6.4 ProBee SE Profiles Description

6.4.1 ProBeeSESI (Energy Service Interface)

The Energy Service Interface connects the energy supply company communication network to the metering and energy management devices within the home. It routes messages to and from the relevant end points. 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).

6.4.1.1 Cluster Support

Table 6-2 ProBeeSESI Cluster Support

Server Side	Client Side
Basic	
Identify	
Time	
Demand Response and Load Control	
Price	
Messaging	
Key establishment	Key establishment

6.4.1.2 Attributes

Table 6-3 ProBeeSESI Attributes

Cluster ID	Cluster name	Attribute ID	Attribute name	Type
0x0000	Basic	0x0000	ZCL version	INT8U
		0x0001	Application version	INT8U
		0x0002	Stack version	INT8U
		0x0003	Hardware version	INT8U
		0x0007	Power source	ENUM8
		0x0011	Physical environment	ENUM8
		0x0012	Device enabled	BOOLEAN
		0x0013	Alarm mask	BITMAP8
		0x0014	Disable local config	BITMAP8
0x0003	Identify	0x0000	Identify time	INT16U

0x000A	Time	0x0000	Time	UTC_TIME
		0x0001	Time Status	INT8U
0x0800	Key establishment (Server Side)	0x0000	Key establishment suite	ENUM16
0x0800	Key establishment (Client Side)	0x0000	Key establishment suite	ENUM16

6.4.1.3 Commands

Table 6-4 ProBeeSESI Commands

Cluster ID	Cluster name	Command ID	Command name	Payload	AT Command
0x0000	Basic (Server-side)	0x00	Reset to Factory Defaults	-	AT+FACTORYRESET
0x0003	Identify (Server-side)	0x00	Identify	2	AT+IDENTIFY
		0x01	Identify Query	-	AT+IDENQUERY
0x0700	Price (Server-side)	0x00	Publish Price	51	AT+PRSETPUBLISH
		0x01	Publish Block Period	17	-
0x0701	Demand Response and Load Control (Server-side)	0x00	Load Control Event	23	AT+DLLOADCTRL
		0x01	Cancel Load Control Event	12	AT+DLCANCELLOAD
		0x02	Cancel All Load Control Events	1	-
0x0703	Messaging (Server-side)	0x00	Display Message	V	AT+MSDISPLAY
		0x01	Cancel Message	2	AT+MSCANCEL
0x0800	Key establishment (Server-side)	0x00	Initiate Key Establishment Request	23	AT+CBKESTART
		0x01	Ephemeral Data Request	22	
		0x02	Confirm Key Data Request	16	
		0x03	Terminate Key Establishment	4	-

6.4.2 ProBeeSEID (In-Premises Display)

The In-Premises Display device will relay 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 should be displayed: current energy usage, a history over selectable periods, pricing information, or text messages. As an interactive device, it can be used for returning simple messages for interpretation by the recipient (e.g. “Button A was pressed”). 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.

6.4.2.1 Cluster Support

Table 6-5 ProBeeSEID Cluster Support

Server Side	Client Side
Basic	Time
Identify	Demand Response and Load Control
	Metering
	Price
	Messaging
Key establishment	Key establishment

6.4.2.2 Attributes

Table 6-6 ProBeeSEID Attributes

Cluster ID	Cluster name	Attribute ID	Attribute name	Type
0x0000	Basic	0x0000	ZCL version	INT8U
		0x0001	Application version	INT8U
		0x0002	Stack version	INT8U
		0x0003	Hardware version	INT8U
		0x0007	Power source	ENUM8
		0x0011	Physical environment	ENUM8
		0x0012	Device enabled	BOOLEAN
		0x0013	Alarm mask	BITMAP8
		0x0014	Disable local config	BITMAP8
0x0003	Identify	0x0000	Identify time	INT16U
0x0701	Demand Response and Load Control	0x0000	Utility enrollment group	INT8U
		0x0001	Start randomize minutes	INT8U
		0x0002	Stop randomize minutes	INT8U
		0x0003	Device class value	INT16U
0x0800	Key establishment (Server Side)	0x0000	Key establishment suite	ENUM16
0x0800	Key establishment (Client Side)	0x0000	Key establishment suite	ENUM16

6.4.2.3 Commands

Table 6-7 ProBeeSEID Commands

Cluster ID	Cluster name	Command ID	Command name	Payload	AT Command
0x0000	Basic (Server-side)	0x00	Reset to Factory Defaults	-	AT+FACTORYRESET
0x0003	Identify (Server-side)	0x00	Identify	2	AT+IDENTIFY
		0x01	Identify Query	-	AT+IDENQUERY
0x0700	Price (Client-side)	0x00	Get Current Price	1	AT+PRGETCUR
		0x01	Get Scheduled Prices	5	AT+PRGETSCH
		0x02	Price Acknowledgement	13	-
		0x03	Get Block Period	5	AT+PRGETBLKPR
0x0701	Demand Response and Load Control (Client-side)	0x00	Report Event Status	60	AT+DLREPORTES
		0x01	Get Scheduled Events	5	AT+DLGETSCH
0x0702	Metering (Client-side)	0x00	Get Profile	6	AT+SMGETPROFD AT+SMGETPROFR
		0x01	Request Mirror Response	2	-
		0x02	Mirror Removed	2	-
		0x03	Request Fast Poll Mode	2	AT+SMFPMODE
0x0703	Messaging (Client-side)	0x00	Get Last Message	-	AT+MSGETLAST
		0x01	Message Confirmation	8	-
0x0800	Key establishment (Server-side)	0x00	Initiate Key Establishment Request	23	AT+CBKESTART
		0x01	Ephemeral Data Request	22	
		0x02	Confirm Key Data Request	16	
		0x03	Terminate Key Establishment	4	-

6.4.3 ProBeeSEME (Metering Device)

The Metering Device 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).

6.4.3.1 Cluster Support

Table 6-8 ProBeeSEME Cluster Support

Server Side	Client Side
Basic	Time
Identify	Messaging
Metering	
Key establishment	Key establishment

6.4.3.2 Attributes

Table 6-9 ProBeeSEME Attributes

Cluster ID	Cluster name	Attribute ID	Attribute name	Type
0x0000	Basic	0x0000	ZCL version	INT8U
		0x0001	Application version	INT8U
		0x0002	Stack version	INT8U
		0x0003	Hardware version	INT8U
		0x0007	Power source	ENUM8
		0x0011	Physical environment	ENUM8
		0x0012	Device enabled	BOOLEAN
		0x0013	Alarm mask	BITMAP8
		0x0014	Disable local config	BITMAP8
0x0003	Identify	0x0000	Identify time	INT16U
0x0702	Metering	0x0000	Current summation delivered	INT48U
		0x0001	Current summation received	INT48U
		0x0002	Current max demand delivered	INT48U
		0x0003	Current max demand received	INT48U
		0x0004	Dft summation	INT48U
		0x0200	Status	BITMAP8

		0x0300	Unit of measure	ENUM8
		0x0303	Summation formatting	BITMAP8
		0x0306	Metering device type	BITMAP8
0x0800	Key establishment (Server Side)	0x0000	Key establishment suite	ENUM16
0x0800	Key establishment (Client Side)	0x0000	Key establishment suite	ENUM16

6.4.3.3 Commands

Table 6-10 ProBeeSEME Commands

Cluster ID	Cluster name	Command ID	Command name	Pay load	AT Command
0x0000	Basic (Server-side)	0x00	Reset to Factory Defaults	-	AT+FACTORYRESET
0x0003	Identify (Server-side)	0x00	Identify	2	AT+IDENTIFY
		0x01	Identify Query	-	AT+IDENQUERY
0x0702	Metering (Server-side)	0x00	Get Profile Response	V	-
		0x01	Request Mirror	-	AT+SMRQMIRR
		0x02	Remove Mirror	-	AT+SMRVMIRR
		0x03	Request Fast Poll Mode Response	5	-
0x0703	Messaging (Client-side)	0x00	Get Last Message	-	AT+MSGETLAST
		0x01	Message Confirmation	8	-
0x0800	Key establishment (Server-side)	0x00	Initiate Key Establishment Request	23	AT+CBKESTART
		0x01	Ephemeral Data Request	22	-
		0x02	Confirm Key Data Request	16	-
		0x03	Terminate Key Establishment	4	-

7 AT Command Reference

7.1 Command Line Format

Each command line is composed of a prefix, body and terminator. Only “AT” can be used for the prefix. The body is a string of characters. The terminator is <CR> or <NL>. A command can be one of the followings:

- Read a configuration parameter:
  AT<Command>?<CR>

- Do something:
  AT<Command><CR>

- Write a configuration parameter:
  AT<Command>=<Parameter 1>,<Parameter 2>,<Parameter 3>,…<CR>

Commands denoted with a “+” character are extended AT commands.
S-registers are supported. Multiple commands in the same command line are not supported.

7.2 Standard AT Commands

7.2.1 AT

Description	Check the state of the connection between host and ProBee device.
Execute	AT<CR>
Response	OK<CR>

7.2.2 ATE

Description	Echo(1) or No Echo(0).
Execute	ATE<number><CR>
Response	OK<CR>

7.2.3 ATR

Description	Response(1) or No Response(0).
Execute	ATR<number><CR>
Response	OK<CR>

7.2.4 ATS

Description	Get/Set the S-registers.		
Execute	ATS<CR>	ATS<number>?<CR>	ATS<number>=<value><CR>

Response	List of the S-register and description	<value><CR> OK<CR>	OK<CR> or ERROR<CR>

7.2.5 ATZ

Description	Reset the node.
Execute	ATZ<CR>
Response	OK<CR>

7.2.6 AT&F

Description	Factory reset.
Execute	AT&F<CR>
Response	OK<CR>

7.2.7 ATU

Description	Get/Set UTC time format type for input/output. (0: 32-bit hexadecimal ranging 0x00000000 ~ 0xFFFFFFFF, 1: MM-DD-YYYY/hh:mm:ss, 2: DD-MM-YYYY/hh:mm:ss, 3: YYYY-MM-DD/hh:mm:ss).	
Execute	ATU?<CR>	ATU=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	1	
Range	0 - 3	

7.3 Node Specific Information

7.3.1 AT+LONGADDR or AT+LA

Description	Get the 64-bit IEEE address. (EUI address, long address) The IEEE address is a unique device address assigned during manufacturing. This address is unique to each physical device.
Execute	AT+LONGADDR?<CR>
Response	<IEEE address><CR> OK<CR>

7.3.2 AT+SHORTADDR or AT+SA

Description	Get the 16-bit Node ID. (network address, short address) The node ID of “0000” is reserved for the coordinator. All other devices receive a randomly generated address from the coordinator or router device that allows the join. This value can change if an address conflict is detected where two devices are found to have the

	same address or a device leaves the network and later joins.
Execute	AT+SHORTADDR?<CR>
Response	<node ID><CR> OK<CR>

7.3.3 AT+ADDRDISCOVERY or AT+AD

Description	Get the corresponding address of a node whose long or short address is known.	
Execute	AT+ADDRDISCOVERY=<IEEE><CR>	AT+ADDRDISCOVERY=<nodeID><CR>
Response	<node ID><CR> OK<CR>	<IEEE address><CR> OK<CR>

7.3.4 AT+OPCH or AT+OC

Description	Get the operating channel number used for transmitting and receiving between RF modules. Uses 802.15.4 channel numbers. A value of "0" means the device has not joined a PAN and is not operating on any channel.
Execute	AT+OPCH?<CR>
Response	<value><CR> OK<CR>

7.3.5 AT+OPPANID or AT+OI

Description	Get the operating 16-bit PAN ID. If PAN ID is set specific value not 0s, operating PAN ID will equal designated PAN ID. A value of "FFFF" means the device is not joined a network.
Execute	AT+OPPANID?<CR>
Response	<value><CR> OK<CR>

7.3.6 AT+OPEPID or AT+OE

Description	Get the operating 64-bit extended PAN ID. A value of "FFFFFFFFFFFFFFFF" means the device is not joined a network.
Execute	AT+OPEPID?<CR>
Response	<value><CR> OK<CR>

7.3.7 AT+VERSION or AT+VR

Description	Get firmware version of the module. The firmware version returns a form of "SExxvx.x".
Execute	AT+VERSION?<CR>
Response	<value><CR>

	OK<CR>

7.3.8 AT+PRODUCTNAME or AT+PN

Description	Get the product name. It can be used to distinguish among different hardware platforms.
Execute	AT+PRODUCTNAME?<CR>
Response	<value><CR> OK<CR>

7.4 Network Specific Parameters

7.4.1 AT+CHMASK or AT+CM

Description	Get/Set the channel mask to use. Bits set as 1 indicate that this particular channel can be used. Bits set as 0 indicate that this particular channel should not be used. For example, a channel mask value of 0x00000001 indicates that only channel 0 should be used. Valid channels range from 11 to 26 inclusive. (16 channels are available.) This translates to a channel mask value of “0x07FFF800”.	
Execute	AT+CHMASK?<CR>	AT+CHMASK=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	0318C800	
Range	00000800 - 07FFF800	

7.4.2 AT+PANID or AT+PI

Description	Get/Set the 16-bit PAN ID. If it is set to 0s at a coordinator, the coordinator will select a random 16-bit hexadecimal number as PAN ID. If it is set to 0s at a router/end-device, the router/end-device will try to join any existing network regardless of PAN ID.	
Execute	AT+PANID?<CR>	AT+PANID=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	0000	
Range	0000 – FFFF	

7.4.3 AT+EPID or AT+EI

Description	Get/Set the 64-bit extended PAN ID. If the node is a coordinator and the EPID is set to 0s, it will create new extended PAN ID using its 64-bit IEEE address as extended PAN ID. If the node is a router or end-device and the EPID is set to 0x, it will try to join any existing network regardless of extended PAN ID.

Execute	AT+EPID?<CR>	AT+EPID=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	0000000000000000	
Range	0000000000000000 – FFFFFFFFFFFFFFFF	

7.4.4 AT+GROUPID or AT+GI

Description	Get/Set the 16-bit Group ID. It is used for multicast data reception.	
Execute	AT+GROUPID?<CR>	AT+GROUPID=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	0000	
Range	0000 – FFFF	

7.4.5 AT+POWER or AT+PW

Description	Get/Set the TX power at which the RF module transmits conducted power.	
Execute	AT+POWER?<CR>	AT+POWER=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	8	
Range	-43 – 8	

7.4.6 AT+STACK or AT+ZS

Description	Get/Set the ZigBee stack profile value. This value must be set the same on all devices that should join the same network. (0: Network specific, 1: ZigBee, 2: ZigBee Pro). Do NOT change default value if you want to use for Smart Energy node.	
Execute	AT+STACK?<CR>	AT+STACK=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	2	
Range	0 – 2	

7.4.7 AT+NODENAME or AT+NN

Description	Get/Set the node name. It can be used to make a clear distinction role or functionality of local node from others.	
Execute	AT+NODENAME?<CR>	AT+NODENAME=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>

Default	ProBee-SExx
Range	Up to 16 characters

7.5 Network Formation and Join

7.5.1 AT+ASCAN or AT+AS

Description	Perform a scanning for existing network. It can take up to 5 seconds. (*CAUTION: It is possible to cause a link error if a node is a parent of an end-device.*)	
Execute	AT+ASCAN<CR>	AT+ASCAN=<channel mask><CR>
Response	<channel>|<PAN ID>|<EPID>|<joinable>|<stack profile> *List of the existing networks …* OK<CR>	

7.5.2 AT+ESCAN or AT+ES

Description	Perform a scanning for energy level of all channels. It can be used to avoid interference. It can take up to 5 seconds. The RSSI energy level ranges from approximately -100 to +127 in dBm. (*CAUTION: It is possible to cause a link error if a node is a parent of an end-device.*)	
Execute	AT+ESCAN<CR>	AT+ESCAN=<channel mask><CR>
Response	<channel>|<RSSI > *List of the RSSI for each channel …* OK<CR>	

7.5.3 AT+DSCAN or AT+DS

Description	Perform a scanning for existing nodes in the current network. A coordinator or router can have children nodes and they are presented by “CLDn”. It can be possible to place some conditions such as scan duration, node type and node name in order. The information of local node is presented by “*”. (*CAUTION: A ProBee-Z*10 with PTv1.2 or earlier cannot receive the response of AT+DSCAN from any ProBee-Z*10 with PTv1.3 or later. It cannot receive from any ProBee-Z*20, either. In this case, it is recommended to upgrade PTv1.2 to the latest version.*)
Execute	AT+DSCAN<CR> or AT+DSCAN=<scan duration><CR> or AT+DSCAN=<scan duration>,<node type><CR> or AT+DSCAN=<scan duration>,<node type>,<node name><CR>
Response	<node type>|<IEEE address>|<node ID>|<version>|<product name>|<node name> *List of the existing nodes and information…* OK<CR>

Range	Scan duration: 1 – 10 Node type: 0 – 4 (Zero indicates all node types.)

7.5.4 AT+RSSI or AT+RS

Description	Get RSSI and LQI information respectively for the delivery of the recent packet across the last hot of the incoming route. The LQI value ranges from 0 to 255 with the maximum value representing the best possible link quality. The RSSI value ranges from approximately -100 to +127 with each value representing the energy level in dBm at the radio's receiver. The sender of the recent packet and the parent are also displayed.
Execute	AT+RSSI?<CR>
Response	Sender: <node ID>, Parent: FFFF, RSSI: <RSSI value>, LQI: <LQI value><CR> OK<CR>

7.5.5 AT+NODETYPE or AT+NT

Description	Get/Set the node type. (0: none, 1: coordinator, 2: router, 3: end-device, 4: sleepy end-device)	
Execute	AT+NODETYPE?<CR>	AT+NODETYPE=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	0	
Range	0 – 4	

7.5.6 AT+PERMIT or AT+PJ

Description	Permit other nodes to join the network. (0~254s, 255=continuance) An end-device cannot perform it.	
Execute	AT+PERMIT<CR>	AT+PERMIT=<value><CR>
Response	OK<CR>	
Range	0 – 255	

7.5.7 AT+LEAVE or AT+LV

Description	Leave the current network.
Execute	AT+LEAVE<CR>
Response	OK<CR>

7.6 End-device

7.6.1 AT+SLEEP or AT+SM

Description	Enter the sleep mode immediately if the node is a sleepy end-device.
Execute	AT+SLEEP<CR>

Response	OK<CR>

7.6.2 AT+PARENTLA or AT+PL

Description	Get the 64-bit IEEE address of the parent node.
Execute	AT+PARENTLA?<CR>
Response	<IEEE address><CR> OK<CR>

7.6.3 AT+PARENTSA or AT+PS

Description	Get the 16-bit Node ID of the parent node.
Execute	AT+PARENTSA?<CR>
Response	<node ID><CR> OK<CR>

7.6.4 AT+CHILDTABLE or AT+CT

Description	Get the child table composed of the end-devices.
Execute	AT+CHILDTABLE?<CR>
Response	<node type>|<IEEE address>|<node ID> *List of the child nodes…* OK<CR>

7.7 Data Transmission and Reception

7.7.1 AT+MAXPAYLOAD or AT+MP

Description	Get the maximum size of the payload that the Application Support sub-layer will accept. The size depends on the security level in use.
Execute	AT+MAXPAYLOAD?<CR>
Response	<value><CR> OK<CR>

7.8 Security

7.8.1 AT+SECURITY or AT+SE

Description	Get/Set the security status. (0: turning security off, 1: on) Every node must be under the same security level in the network. Do NOT change default value if you want to use for Smart Energy node.	
Execute	AT+SECURITY?<CR>	AT+SECURITY=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	1	
Range	0 – 1	

7.8.2 AT+NWKKEY or AT+NK

Description	Get/Set the 128-bit network key. (0 or 0s: generate random 128-bit number)	
Execute	AT+NWKKEY?<CR>	AT+NWKKEY=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF	

7.8.3 AT+UPDATEKEY or AT+UK

Description	Update the network key with a new generated key. (possible only for coordinator) It is applied after 30s later because sending too quickly may cause a sleepy end-device to miss the alternate encryption key and only get the switch key message, which means it will be unable to change to the new network key.
Execute	AT+UPDATEKEY<CR>
Response	OK<CR> or ERROR<CR>

7.8.4 AT+KEYTABLE or AT+KT

Description	Get/Set/Remove link key table.		
Execute	AT+KEYTABLE?<CR>	AT+KEYTABLE=<num>,<IEEE address>,<link key><CR>	AT+KEYTABLE=<num><CR>
Response	<list of the link key table>	OK<CR> or ERROR<CR>	KEY TABLE <num> REMOVED<CR> OK<CR> or ERROR<CR>

7.9 UART

7.9.1 AT+BAUDRATE or AT+UB

Description	Get/Set the serial port speed for communication between the module serial port and host.	
Execute	AT+BAUDRATE?<CR>	AT+BAUDRATE=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	9600	
Range	300 – 230400	

7.9.2 AT+DATABIT or AT+UD

Description	Get/Set the serial data bit setting on the module.	
Execute	AT+DATABIT?<CR>	AT+DATABIT=<value><CR>

Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	8	
Range	7 – 8	

7.9.3 AT+PARITY or AT+UP

Description	Get/Set the serial parity setting on the module.	
Execute	AT+PARITY?<CR>	AT+PARITY=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	0	
Range	0 – 2	

7.9.4 AT+STOPBIT or AT+US

Description	Get/Set the serial stop bit setting on the module.	
Execute	AT+STOPBIT?<CR>	AT+STOPBIT=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	1	
Range	1 – 2	

7.9.5 AT+FLOWCTRL or AT+UF

Description	Get/Set flow control status. (0: none, 1: software, 2: hardware flow control)	
Execute	AT+FLOWCTRL?<CR>	AT+FLOWCTRL=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	0	
Range	0 – 2	

7.10 GPIO

7.10.1 AT+GPIO or AT+IO

Description	Get/Set the configuration status of GPIO individually or all at once. Only GPIO9-14 can be set 0-5, others are available 0-3 and 5.	
Execute	AT+GPIO?<CR> or AT+GPIO<number>?<CR>	AT+GPIO=<values><CR> or AT+GPIO<number>=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	55555000000000055	
Range	GPIO9-14: 0-6, the others: 0-3 and 5-6	

	0	Disabled (unmonitored digital input)
	1	Digital input, monitored
	2	Digital output, default low
	3	Digital output, default high
	4	Analog input, single ended (only valid for GPIO9~14)
	5	Reserved for pin-specific alternate functionalities

7.10.2 AT+DIO

Description	Get/Set the digital IO value individually or all at once. All GPIO pins can be used as digital IO and it is only possible to set 0 or 1. If there are some pins used analog input, they are displayed by "*".	
Execute	AT+DIO?<CR> or AT+DIO<number>?<CR>	AT+DIO=<values><CR> or AT+DIO<number>=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	00000000000000000	
Range	00000000000000000 – 11111111111111111	

7.10.3 AT+AI

Description	Get the analog input value individually or all at once. GPIO9~14 are possible to set ADC and it's read only. If there are some pins used digital IO, they are displayed by "****".	
Execute	AT+AI?<CR>	AT+AI<number>?<CR>
Response	<value>,<value>,<value>,<value><CR> OK<CR>	<value><CR> OK<CR>

7.10.4 AT+ SOURCELA or AT+SL

Description	Get/Set the 64-bit source IEEE address for GPIO remote control. It is possible to limit on the implementation of this functionality by setting specific address. If set to default value, "FFFFFFFFFFFFFFFF", the node allows that any node can control the GPIO status.	
Execute	AT+SOURCELA?<CR>	AT+SOURCELA=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	FFFFFFFFFFFFFFFF	
Range	0000000000000000 – FFFFFFFFFFFFFFFF	

7.11 Firmware Upload and Help

7.11.1 AT+BOOTLOAD or AT+BL

Description	Start bootloader to upload firmware. (blank: local, 1: passthru) The device leaves the AT command line and enters the bootloader menu for uploading new ProBee firmware. There's no restrict to use local upload but a sleepy end-device cannot upload to remote node over the air. The current UART should be set 115200bps, 8 data bits, no parity, 1 stop bit and none flow control.	
Execute	AT+BOOTLOAD<CR>	AT+BOOTLOAD=<number>,<IEEE><CR>
Response	*Bootloader menu* … Serial upload complete	Please start .ebl upload image ...<CR> or Start cloning ...<CR> *Bootloader menu* … Bootload Complete!

7.11.2 AT+HELP or AT+HP

Description	Show all the available commands
Execute	AT+HELP<CR>
Response	Available AT Commands … OK<CR>

7.12 Smart Energy Commands

7.12.1 AT+ATTR or AT+AR

Description	Show all attributes of local node.	
Execute	AT+ATTR?<CR>	AT+ATTR=<clusterID><CR>
Response	clus|attr|data(cluster name) <clusterID>|<attributeID>|<data>(<clusterName>) … OK<CR>	

7.12.2 AT+INFO or AT+IF

Description	Show Smart Energy device information
Execute	AT+INFO?<CR>
Response	Security Level: <security level>, Security Profile: SE End-point count: <endpointCount> End-point No.<endpoint>: ProfileID-<profileID>, DeviceID-<deviceID> Client-side cluster: <clusterID>(<clusterName>) Server-side cluster: <clusterID>(<clusterName>) ...

	OK<CR>

7.12.3 AT+DEST or AT+DE

Description	Set address of destination node. This address will be used in AT+CREAD, AT+CWRITE, AT+CCMD, AT+SCMD and other SE commands.
Execute	AT+DEST=<nodeID>,<endpoint><CR> or AT+DEST=<IEEE address>,<endpoint><CR>
Response	OK<CR> or ERROR<CR>

7.12.4 AT+SREAD or AT+SR

Description	Read attribute from local node. (server-side command)
Execute	AT+SREAD=<clusterID>,<attributeID><CR>
Response	<attribute value> OK<CR> or ERROR<CR>

7.12.5 AT+CREAD or AT+CR

Description	Read attribute from remote node. (client-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+CREAD=<clusterID>,<attributeID><CR>
Response	<attribute value> OK<CR> or ERROR<CR>

7.12.6 AT+SWRITE or AT+SW

Description	Write attribute to local node. (server-side command)
Execute	AT+SWRITE=<clusterID>,<attributeID>,<value><CR>
Response	OK<CR> or ERROR<CR>

7.12.7 AT+CWRITE or AT+CW

Description	Write attribute to remote node. (client-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+CWRITE=<clusterID>,<attributeID>,<value><CR>
Response	OK<CR> or ERROR<CR>

7.12.8 AT+SCMD or AT+SC

Description	Send server-side command message to destination node. (server-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+SCMD=<clusterID>,<commandID><CR>
Response	OK<CR> or ERROR<CR>

7.12.9 AT+CCMD or AT+CC

Description	Send client-side command message to destination node. (client-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+CCMD=<clusterID>,<commandID><CR>
Response	OK<CR> or ERROR<CR>

7.12.10 AT+FACTORYRESET or AT+FR

Description	Send 'Factory Reset' command(Basic cluster) to destination node. (client-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+FACTORYRESET?<CR>
Response	OK<CR> or ERROR<CR>

7.12.11 AT+IDENTIFY or AT+II

Description	Send 'Identify' command(Identify cluster) to destination node. (client-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+IDENTIFY=<time><CR>
Response	OK<CR> or ERROR<CR>

7.12.12 AT+IDENQUERY or AT+IQ

Description	Send 'Identify Query' command(Identify cluster) to destination node. (client-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+IDENQUERY?<CR>
Response	OK<CR> or ERROR<CR>

7.12.13 AT+TIME or AT+TM

Description	Get/Set Global Time. The format of time is determined by ATU value.	
Execute	AT+TIME?<CR>	AT+TIME=<UTC time>
Response	<UTC time><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	01-01-2000/00:00:00	
Range	01-01-2000/00:00:00 - 02-07-2136/06:28:15	

7.12.14 AT+TSYNC or AT+TS

Description	Read current time data from server-side Time Cluster (server-side Time Cluster)
Execute	AT+DEST=<address>,<endpoint><CR>

	AT+TSYNC<CR>
Response	+<address>|SE_GLOBAL_COMMAND|000A|01 +<address>|SE_TIME_SYNC|<value><CR> <value><CR>

7.12.15 AT+CBKESTART or AT+CS

Description	Start 'Initiate Key Establishment Request' command(Key Establishment cluster) to destination node and exchange static and ephemeral data. This command should be performed at Trust Center(coordinator) node or destination node should be Trust Center(coordinator) node.	
Execute	AT+DEST=<address>,<endpoint><CR> AT+CBKESTART<CR>	AT+CBKESTART=<address>,<endpoint><CR>
Response	CBKE DEST: <address>, EP: <endpoint>, RESULT: <result ID><CR> OK<CR> or ERROR<CR>	

7.12.16 AT+CBKEPARTNER or AT+CP

Description	Set destination node as CBKE partner node for exchanging partner link key. This command should not be performed at Trust Center(coordinator) node nor destination node should not be Trust Center(coordinator) node.	
Execute	AT+DEST=<address>,<endpoint><CR> AT+CBKEPARTNER<CR>	AT+CBKEPARTNER=<address>,<endpoint><CR>
Response	CBKE DEST: <address>, EP: <endpoint>, RESULT: <result ID><CR> OK<CR> or ERROR<CR>	

7.12.17 AT+IMPLICIT or AT+IM

Description	Get/Set CBKE Data Token - Device Implicit Cert.	
Execute	AT+IMPLICIT?<CR>	AT+IMPLICIT=<value><CR>
Response	CBKE DATA TOKEN IS EMPTY<CR> or CBKE DATA TOKEN IS LOCKED<CR> DEVICE IMPLICIT CERT: <value> CA PUBLIC KEY & DEVICE PRIVATE KEY: <value><CR> OK<CR>	

7.12.18 AT+PUBLICKEY or AT+PK

Description	Get/Set CBKE Data Token - CA Public Key & Device Private Key	
Execute	AT+PUBLICKEY?<CR>	AT+PUBLICKEY=<value><CR>
Response	CBKE DATA TOKEN IS EMPTY<CR> or CBKE DATA TOKEN IS LOCKED<CR> DEVICE IMPLICIT CERT: <value>	

	CA PUBLIC KEY & DEVICE PRIVATE KEY: <value><CR> OK<CR>

7.12.19 AT+SETCBKEKEY

Description	Lock CBKE Data Token. Once the CBKE Data Token is locked, it cannot be changed and erased.
Execute	AT+SETCBKEKEY<CR>
Response	CBKE DATA TOKEN IS LOCKED<CR> DEVICE IMPLICIT CERT: <value> CA PUBLIC KEY & DEVICE PRIVATE KEY: <value><CR> OK<CR>

7.13 ProBeeSESI (Energy Service Interface) Profile Commands

7.13.1 AT+PRSETPUBLISH or AT+PSP

Description	Send 'Publish Price' command(Price cluster) to destination node. (server-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+PRSETPUBLISH=<providerID>,<issuerID>,<unit of measure>,<currency>,<price tier>,<number of price tiers>,<duration in minites>,<price>,<price ratio><CR>
Response	OK<CR> or ERROR<CR>

7.13.2 AT+PRLABEL or AT+PLB

Description	Get/Set Price Label. This value will be used in AT+PRSETPUBLISH.	
Execute	AT+PRLABEL?<CR>	AT+PRLABEL=<value><CR>
Response	<value> OK<CR>	OK<CR> or ERROR<CR>

7.13.3 AT+PRCURRENTTIME or AT+PCT

Description	Get/Set Price Current Time. This value will be used in AT+PRSETPUBLISH. The format of time is determined by ATU value.	
Execute	AT+PRCURRENTTIME?<CR>	AT+PRCURRENTTIME=<UTC time>
Response	<UTC time><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	01-01-2000/00:00:00	
Range	01-01-2000/00:00:00 - 02-07-2136/06:28:15	

7.13.4 AT+PRSTARTTIME or AT+PST

Description	Get/Set Price Start Time. This value will be used in AT+PRSETPUBLISH. The

	format of time is determined by ATU value.	
Execute	AT+PRSTARTTIME?<CR>	AT+PRSTARTTIME=<UTC time>
Response	<UTC time><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	01-01-2000/00:00:00	
Range	01-01-2000/00:00:00 - 02-07-2136/06:28:15	

7.13.5 AT+MSID or AT+MID

Description	Get/Set Message ID. This value will be used in AT+MSDISPLAY and AT+MSCANCEL.	
Execute	AT+MSID?<CR>	AT+MSID=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	00000000	
Range	00000000 – FFFFFFFF	

7.13.6 AT+MSCTRL or AT+MCT

Description	Get/Set Message Control. This value will be used in AT+MSDISPLAY and AT+MSCANCEL.	
Execute	AT+MSCTRL?<CR>	AT+MSCTRL=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	00	
Range	00 – FF	

7.13.7 AT+MSSTARTTIME or AT+MST

Description	Get/Set Messaging Start Time. This value will be used in AT+MSDISPLAY. The format of time is determined by ATU value.	
Execute	AT+MSSTARTTIME?<CR>	AT+MSSTARTTIME=<UTC time>
Response	<UTC time><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	01-01-2000/00:00:00	
Range	01-01-2000/00:00:00 - 02-07-2136/06:28:15	

7.13.8 AT+MSDURATION or AT+MDU

Description	Get/Set Duration in Minutes. This value will be used in AT+MSDISPLAY.	
Execute	AT+MSDURATION?<CR>	AT+MSDURATION=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>

Default	0000
Range	0000 – FFFF

7.13.9 AT+MSDISPLAY or AT+MDI

Description	Send 'Display Message' command(Messaging cluster) to destination node. (server-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+MSDISPLAY=<data>?<CR>
Response	OK<CR> or ERROR<CR>

7.13.10 AT+MSCANCEL or AT+MCA

Description	Send 'Cancel Message' command(Messaging cluster) to destination node. (server-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+MSCANCEL?<CR>
Response	OK<CR> or ERROR<CR>

7.13.11 AT+DLLIST or AT+DLT

Description	Show DRLC Event List.
Execute	AT+DLLIST?<CR>
Response	event ID|clss|gr|dura|cr|cool(sp)|heat(sp)|av|du|ct|start time .. OK<CR>

7.13.12 AT+DLSTARTTIME or AT+DST

Description	Get/Set DRLC Start Time. This value will be used in AT+DLSET. The format of time is determined by ATU value.	
Execute	AT+DLSTARTTIME?<CR>	AT+DLSTARTTIME=<UTC time>
Response	<UTC time><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	01-01-2000/00:00:00	
Range	01-01-2000/00:00:00 - 02-07-2136/06:28:15	

7.13.13 AT+DLSET or AT+DLS

Description	Add/Modify DRLC Event.
Execute	AT+DLLIST=<event ID>,<device class>,<utility enrolment group>,<duration in minutes>,<criticality level>,<cooling offset>,<cooling set point>,<heating offset>,<heating set point>,<average load adjustment percentage>,<duty cycle>,<event control><CR>

Response	OK<CR> or ERROR<CR>

7.13.14 AT+DLDEL or AT+DLD

Description	Delete DRLC Event.
Execute	AT+DLDEL=<event ID><CR>
Response	OK<CR> or ERROR<CR>

7.13.15 AT+DLLOADCTRL or AT+DLC

Description	Send 'Load Control Event' command(DRLC cluster) to destination node. (server-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+DLLOADCTRL=<event ID><CR>
Response	OK<CR> or ERROR<CR>

7.13.16 AT+DLCANCELLOAD or AT+DCL

Description	Send 'Cancel Load Control Event' command(DRLC cluster) to destination node. (server-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+DLCANCELLOAD=<event ID><CR>
Response	OK<CR> or ERROR<CR>

7.14 ProBeeSEID (In-Premises Display) Profile Commands

7.14.1 AT+SMGETPROFD or AT+SGD

Description	Send 'Get Profile' command(Metering cluster) with 'Consumption Delivered' to destination node. (client-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+SMGETPROFD=<number of periods><CR>
Response	OK<CR> or ERROR<CR>

7.14.2 AT+SMGETPROFR or AT+SGR

Description	Send 'Get Profile' command(Metering cluster) with 'Consumption Received' to destination node. (client-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+SMGETPROFR=<number of periods><CR>
Response	OK<CR> or ERROR<CR>

7.14.3 AT+SMFPMODE or AT+SRF

Description	Send 'Request Fast Poll Mode' command(Metering cluster) to destination node. (client-side command)

Execute	AT+DEST=<address>,<endpoint><CR> AT+SMFPMODE=<duration>,<period><CR>
Response	OK<CR> or ERROR<CR>

7.14.4 AT+SMENDTIME or AT+SET

Description	Get/Set Metering End Time. This value will be used in AT+SMGETPROFD and AT+SMGETPROFR. The format of time is determined by ATU value.	
Execute	AT+SMENDTIME?<CR>	AT+SMENDTIME=<UTC time>
Response	<UTC time><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	01-01-2000/00:00:00	
Range	01-01-2000/00:00:00 - 02-07-2136/06:28:15	

7.14.5 AT+PRGETCUR or AT+PGC

Description	Send 'Get Current Price' command(Price cluster) to destination node. (client-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+PRGETCUR=<command options><CR>
Response	OK<CR> or ERROR<CR>
Range	0 - 1

7.14.6 AT+PRGETSCH or AT+PGS

Description	Send 'Get Scheduled Prices' command(Price cluster) to destination node. (client-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+PRGETSCH=<number of events><CR>
Response	OK<CR> or ERROR<CR>

7.14.7 AT+PRGETBLKPR or AT+PGB

Description	Send 'Get Block Period(s)' command(Price cluster) to destination node. (client-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+PRGETBLKPR=<number of events><CR>
Response	OK<CR> or ERROR<CR>

7.14.8 AT+PRSTARTTIME or AT+PST

Description	Get/Set Price Start Time. This value will be used in AT+PRGETSCH and AT+PRGETBLKPR. The format of time is determined by ATU value.	
Execute	AT+PRSTARTTIME?<CR>	AT+PRSTARTTIME=<UTC time>

Response	<UTC time><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	01-01-2000/00:00:00	
Range	01-01-2000/00:00:00 - 02-07-2136/06:28:15	

7.14.9 AT+MSGETLAST or AT+MGL

Description	Send 'Get Last Message' command(Messaging cluster) to destination node. (client-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+MSGETLAST<CR>
Response	OK<CR> or ERROR<CR>

7.14.10 AT+DLLIST or AT+DLT

Description	Show DRLC Event List.
Execute	AT+DLLIST?<CR>
Response	event ID|clss|gr|dura|cr|cool(sp)|heat(sp)|av|du|ct|start time .. OK<CR>

7.14.11 AT+DLSTARTTIME or AT+DST

Description	Get/Set DRLC Start Time. This value will be used in AT+DLUPDATE and AT+DLGETSCH. The format of time is determined by ATU value.	
Execute	AT+DLSTARTTIME?<CR>	AT+DLSTARTTIME=<UTC time>
Response	<UTC time><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	01-01-2000/00:00:00	
Range	01-01-2000/00:00:00 - 02-07-2136/06:28:15	

7.14.12 AT+DLUPDATE or AT+DLU

Description	Update DRLC Event.
Execute	AT+DLUPDATE=<event ID>,<criticality level>,<cooling set point>,<heating set point>,<average load adjustment percentage>,<duty cycle>,<event control><CR>
Response	OK<CR> or ERROR<CR>

7.14.13 AT+DLREPORTES or AT+DRE

Description	Send 'Report Event Status' command(DRLC cluster) to destination node. (client-side command)
Execute	AT+DEST=<address>,<endpoint><CR>

	AT+DLREPORTES=<event ID>,<event status><CR>
Response	OK<CR> or ERROR<CR>

7.14.14 AT+DLGETSCH or AT+DGS

Description	Send 'Get Scheduled Events' command(DRLC cluster) to destination node. (client-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+DLGETSCH=<number of events ><CR>
Response	OK<CR> or ERROR<CR>

7.15 ProBeeSEME (Metering Device) Profile Commands

7.15.1 AT+SMRQMIRR or AT+SQM

Description	Send 'Request Mirror Command' command(Metering cluster) to destination node. (server-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+SMRQMIRR<CR>
Response	OK<CR> or ERROR<CR>

7.15.2 AT+SMRVMIRR or AT+SVM

Description	Send 'Remove Mirror Command' command(Metering cluster) to destination node. (server-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+SMRVMIRR<CR>
Response	OK<CR> or ERROR<CR>

7.15.3 AT+SMSETPR or AT+SSP

Description	Get/Set Metering Profile. This value will be used in 'Get Profile Response' command.	
Execute	AT+SMSETPR?<CR>	AT+SMSETPR=<interval period>,<metering interval 1>,<metering interval>,...<CR>
Response	ProfileIntervalPeriod:<interval period>, NumberOfPeriodsDelivered: <number of periods delivered><CR> <metering interval 1><CR> <metering interval 2><CR> … OK<CR>	OK<CR> or ERROR<CR>

7.15.4 AT+SMFPENDTIME or AT+SFT

Description	Get/Set Metering Fast Poll Mode End Time. This value will be used in 'Request Fast Poll Mode Response' command. The format of time is determined by ATU value.	
Execute	AT+SMFPENDTIME?<CR>	AT+SMFPENDTIME=<UTC time>
Response	<UTC time><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	01-01-2000/00:00:00	
Range	01-01-2000/00:00:00 - 02-07-2136/06:28:15	

7.15.5 AT+SMFPPERIOD or AT+SFP

Description	Get/Set Metering Fast Poll Mode Period. This value will be used in 'Get Profile Response' command.	
Execute	AT+SMFPPERIOD?<CR>	AT+SMFPPERIOD=<value><CR>
Response	<value><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	00	
Range	00 - FF	

7.15.6 AT+SMENDTIME or AT+SET

Description	Get/Set Metering End Time. This value will be used in 'Get Profile Response' command. The format of time is determined by ATU value.	
Execute	AT+SMENDTIME?<CR>	AT+SMENDTIME=<UTC time>
Response	<UTC time><CR> OK<CR>	OK<CR> or ERROR<CR>
Default	01-01-2000/00:00:00	
Range	01-01-2000/00:00:00 - 02-07-2136/06:28:15	

7.15.7 AT+MSGETLAST or AT+MGL

Description	Send 'Get Last Message' command(Messaging cluster) to destination node. (client-side command)
Execute	AT+DEST=<address>,<endpoint><CR> AT+MSGETLAST<CR>
Response	OK<CR> or ERROR<CR>

8 S-Registers

Register number	Descriptions
11	Display incoming message in command mode
12	Display command echo
13	Display command responses
14	Manage LED status
22	Permit joining timeout
23	TX power mode
41	GPIO pull-up
42	GPIO monitoring period
43	GPIO monitoring node
44	GPIO change detection sampling
45	GPIO remote control
51	Poll timeout for a parent node
52	Poll period for a sleepy end-device
53	Sleep duration
54	Standby timeout
55	Rejoin timeout
56	Poll period for an end-device
71	Enable Frequency agility

8.1 Display Options

8.1.1 S11

Description	Get/Set the display mode. It determines to display incoming message in command mode.	
Execute	ATS11?<CR>	ATS11=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	1	
Range	0 - 1	

8.1.2 S12

Description	Get/Set the command echo mode.	
Execute	ATS12?<CR>	ATS12=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	1	
Range	0 – 1	

8.1.3 S13

Description	Get/Set the command responses mode.	
Execute	ATS13?<CR>	ATS13=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	1	
Range	0 - 1	

8.1.4 S14

Description	Get/Set the LED mode. It causes the LED operation if the device is ZE20S set GPIO15, 16 to specific functionality. 0: LEDs are always OFF regardless of configuration or status. 1: LEDs are OFF when the node is under sleep status. 2: LEDs always operate by the case.	
Execute	ATS14?<CR>	ATS14=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	1	
Range	0 - 2	

8.2 Network Settings

8.2.1 S22

Description	Get/Set the timeout to permit joining. It determines the time in seconds that the device will allow other devices to join to it when it starts network but an end-device cannot perform it. If set to 255, the device will always permit joining.	
Execute	ATS22?<CR>	ATS22=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	255	
Range	0 - 255	

8.2.2 S23

Description	Get/Set the power boost mode of the module. If enabled, boost mode improves sensitivity and increases output power, improving the link margin and range.	
Execute	ATS23?<CR>	ATS23=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>

Default	0
Range	0 – 1

8.3 GPIO

8.3.1 S41

Description	Get/Set the GPIO pull up/down option. If GPIOs are set to DI(Digital Input), it is possible to pull-up or pull-down with this value. Pull-down = 0, Pull-up = 1.	
Execute	ATS41?<CR>	ATS41=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	0	
Range	0 - 1	

8.3.2 S42

Description	Get/Set the GPIO monitoring period in seconds. If it is enabled, a node sends GPIO values to a remote node based on preconfigured destination IEEE address periodically and it is possible to monitor the GPIO status at the remote node. A remote node can also be the sink node according to GPIO monitoring node (S43). If set to 0, GPIO monitoring is disabled.	
Execute	ATS42?<CR>	ATS42=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	0	
Range	0 – 10000	

8.3.3 S43

Description	Get/Set the GPIO monitoring node. If S43 is 0, the monitoring node is selected by AT+DESTLA or AT+DL command. If S43 is 1, the sink node is set as the monitoring node.	
Execute	ATS43?<CR>	ATS43=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	0	
Range	0 – 10000	

8.3.4 S44

Description	Get/Set the GPIO change detection sampling mode. If it is enabled, a node sends GPIO values to a remote node based on preconfigured destination IEEE address immediately when the status of GPIO changes. A remote node can also be the sink

	node according to GPIO monitoring node (S43). If set to 0, GPIO change detection sampling is disabled.	
Execute	ATS44?<CR>	ATS44=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	0	
Range	0 – 1	

8.3.5 S45

Description	Get/Set the GPIO remote control mode. If any GPIO pin is set to digital input, a node can control the I/O status of a remote node if the I/O pin of the remote node is set to digital output. It is possible to use when GPIO monitoring is enabled (GPIO monitoring period, S42 > 0) or GPIO change detection sampling is enabled.(GPIO change detection sampling mode, S44 = 1)	
Execute	ATS45?<CR>	ATS45=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	0	
Range	0 – 1	

8.4 End-Device

8.4.1 S51

Description	Get/Set the poll timeout for a parent node in seconds. If an end-device or a sleepy end-device does not send a poll request to its parent coordinator or router within the poll timeout, the end-device is removed from the child table.	
Execute	ATS51?<CR>	ATS51=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	60	
Range	0 – 4177920	

8.4.2 S52

Description	Get/Set the sleepy end-device poll period in milliseconds. Adaptive polling may allow the sleepy end-device to poll more rapidly for a short time when receiving RF data.	
Execute	ATS52?<CR>	ATS52=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	100	
Range	0 - 10000	

8.4.3 S53

Description	Get/Set the sleep duration in quarter seconds. An end-device sleeps for this time and turns off radio every sleep cycle. If a node sleeps over poll timeout of parent node, the device will be removed from the child table.	
Execute	ATS53?<CR>	ATS53=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	8	
Range	0 – 16711680	

8.4.4 S54

Description	Get/Set the standby timeout in seconds. When an end-device wakes up and polls with the parent node, the node will wait for another data from the ZigBee network or from the serial port for S54 before it goes to sleep mode again. If any serial or RF data is sent or received, the timer will reset and wait for next again.	
Execute	ATS54?<CR>	ATS54=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	5	
Range	0 – 10000	

8.4.5 S55

Description	Get/Set the rejoin timeout. If it is enabled, an end-device tries to rejoin the current network when contact with the network has been lost for setting value in seconds. The most common usage case is when an end-device can no longer communicate with its parent and wishes to find a new one. A value of zero, do not try to rejoin.	
Execute	ATS55?<CR>	ATS55=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	10	
Range	0 – 10000	

8.4.6 S56

Description	Get/Set the end-device poll period in seconds. Adaptive polling may allow the end-device to poll more rapidly for a short time when receiving RF data.	
Execute	ATS56?<CR>	ATS56=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>

Default	5
Range	0 - 10000

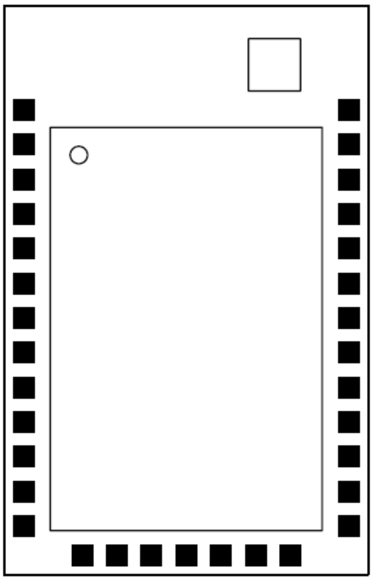
8.5 Frequency Agility

8.5.1 S71

Description	Get/Set the Frequency agility function. If it is enabled in coordinator mode, node will automatically shift operating channel to one of channel masks when its unicasts fail more than threshold.	
Execute	ATS71?<CR>	ATS71=<value><CR>
Response	<value><CR> OK<CR>	OK<CR>
Default	0	
Range	0 – 1	

9 Pin Assignments

NO	Pin Assignment
1	GND
2	PC_5/FACTORY_RST
3	PC_6/GPIO0
4	PC_7/DTR/GPIO1
5	PA_7/DSR/GPIO2
6	PB_3/CTS/GPIO3
7	PB_4/RTS/GPIO4
8	PA_0/MOSI/GPIO5
9	PA_1/MISO/GPIO6
10	PA_2/SCLK/GPIO7
11	PA_3/nSSEL/GPIO8
12	VCC
13	GND



Pin Assignment	NO
GND	33
PB_5/ADC0/GPIO9	32
PB_6/ADC1/GPIO10	31
PB_7/ADC2/GPIO11	30
PC_1/ADC3/GPIO12	29
PA_4/ADC4/GPIO13	28
PA_5/ADC5/GPIO14	27
PB_1/TXD	26
PB_2/RXD	25
PB_0/GPIO15	24
PA_6/GPIO16	23
/RESET	22
GND	21

VCC	JTCK	PC_2/JTDO	PC_3/JTDI	PC_4/JTMS	PC_0/JRST	GND
14	15	16	17	18	19	20

Figure 10-1 Pin Assignments

Table 10-1 Pin Assignments Table

Pin	NAME	Default Function	Direction	Description
1	GND	-	-	Ground
2	Factory Reset	Factory Reset	IN	Digital I/O, Factory reset input, Active low
3	GPIO_0	Permit Joining	IN/OUT	Digital I/O, Permit joining input
4	GPIO_1	UART_DTR	IN/OUT	Digital I/O, UART_DTR
5	GPIO_2	UART_DSR	IN/OUT	Digital I/O, UART_DSR
6	GPIO_3	UART_CTS	IN/OUT	Digital I/O, UART_CTS
7	GPIO_4	UART_RTS	IN/OUT	Digital I/O, UART_RTS
8	GPIO_5	DIO_5	IN/OUT	Digital I/O
9	GPIO_6	DIO_6	IN/OUT	Digital I/O
10	GPIO_7	DIO_7	IN/OUT	Digital I/O
11	GPIO_8	DIO_8	IN/OUT	Digital I/O
12	VCC	-	IN	Power supply, 3.3V
13	GND	-	-	Ground
14	VCC	-	-	Power supply, 3.3V
15	JTCK	-	IN	JTAG clock input from debugger
16	JTDO	-	OUT	JTAG data output to debugger
17	JTDI	-	IN	JTAG data input from debugger
18	JTMS	-	IN	JTAG mode select from debugger
19	JRST	-	IN	JTAG reset input from debugger
20	GND	-	-	Ground
21	GND	-	-	Ground
22	/RESET	-	IN	H/W_/Reset, Active low
23	GPIO_16	Status LED	IN/OUT	Digital I/O, Status LED
24	GPIO_15	Power LED	IN/OUT	Digital I/O, Power LED
25	UART_RXD	UART_RXD	IN	UART Data Input
26	UART_TXD	UART_TXD	OUT	UART Data Output
27	GPIO_14	DIO_14	IN/OUT	Digital I/O / ADC_5
28	GPIO_13	DIO_13	IN/OUT	Digital I/O / ADC_4
29	GPIO_12	DIO_12	IN/OUT	Digital I/O / ADC_3
30	GPIO_11	DIO_11	IN/OUT	Digital I/O / ADC_2
31	GPIO_10	DIO_10	IN/OUT	Digital I/O / ADC_1
32	GPIO_9	DIO_9	IN/OUT	Digital I/O / ADC_0
33	GND	-	-	Ground

10 Connections

10.1 Serial Port Pin Outs

ProBee-ZE20S is a DCE device compatible with the RS232 standard, a DB9 female interface.

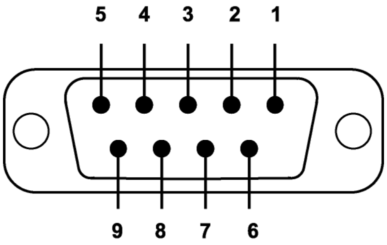


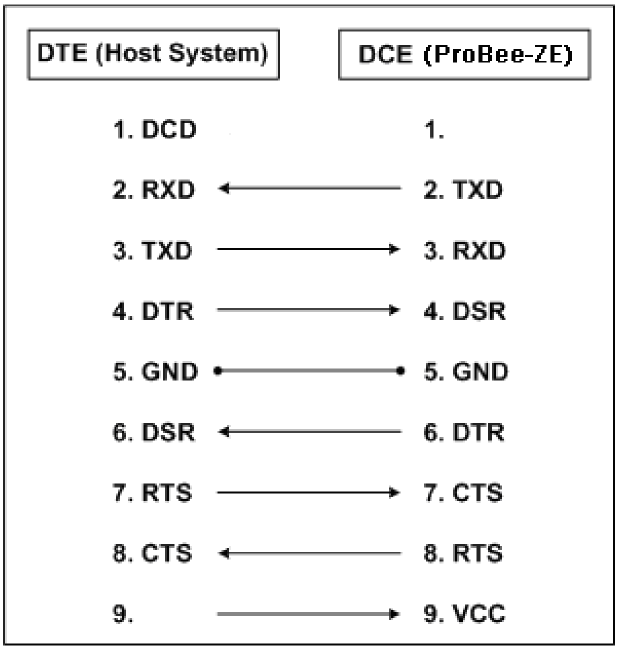
Figure 11-1 Pin layout of the DB-9 female connector

Table 11-1 Pin assignment of the DB-9 female connector

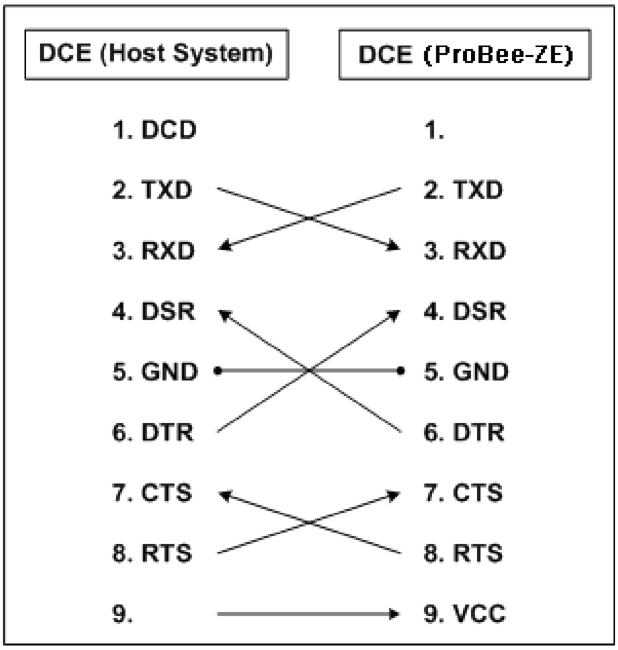
Pin #	Signal	Direction	Description
1	-	-	-
2	TxD	Output	Transmitted Data
3	RxD	Input	Received Data
4	DSR	Input	Data set ready
5	GND	-	Signal Ground
6	DTR	Output	Data terminal ready
7	CTS	Input	Clear to Send
8	RTS	Output	Ready to send
9	VCC	Input	(5V ~ 12V)

10.2 Serial Wiring Diagram

10.2.1 To Host with DTE Interface



10.2.2 To Host with DCE Interface



11 Mechanical Drawings

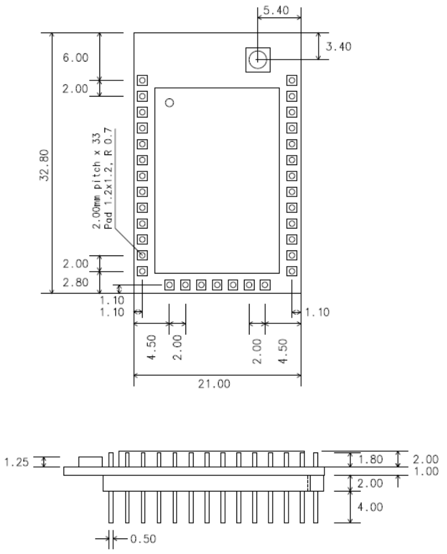


Figure 12-1 ProBee-ZE20SDU

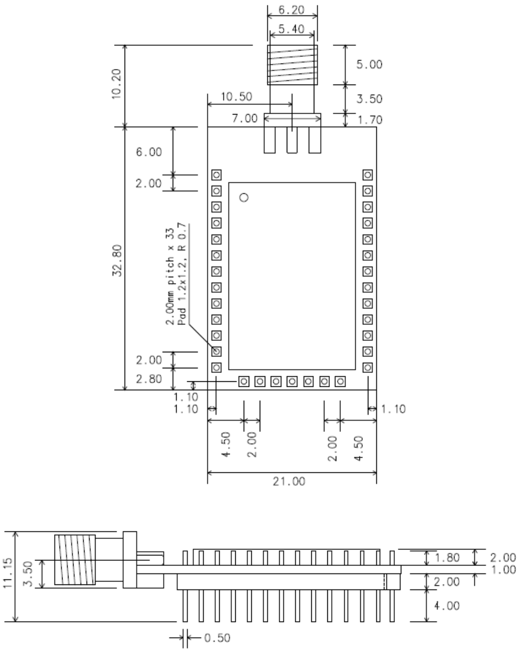


Figure 12-2 ProBee-ZE20SDS

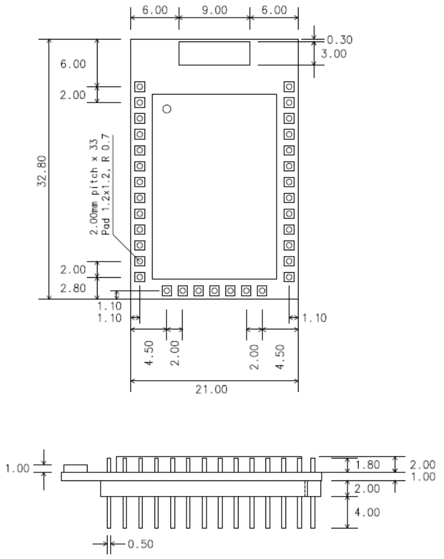


Figure 12-3 ProBee-ZE20SDC

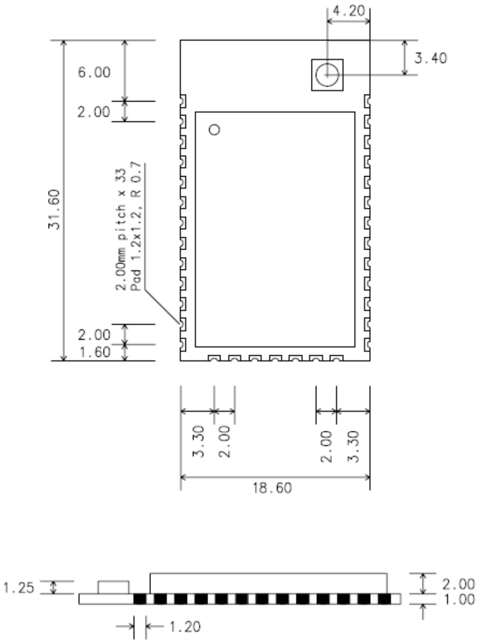


Figure 12-4 ProBee-ZE20SSU

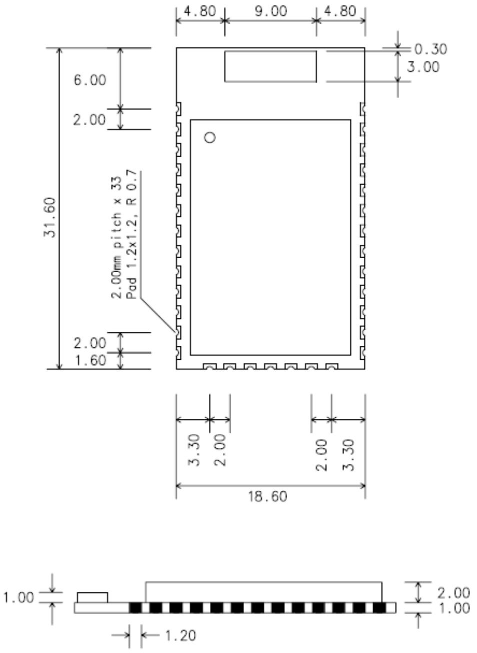


Figure 12-5 ProBee-ZE20SSC

12 Warranty

12.1 GENERAL WARRANTY POLICY

Sena Technologies, Inc. (hereinafter referred to as Sena) warrants that the Product shall conform to and perform in accordance with published technical specifications and the accompanying written materials, and shall be free of defects in materials and workmanship, for the period of time herein indicated, such warranty period commencing upon receipt of the Product.

This warranty is limited to the repair and/or replacement, at Sena's discretion, of defective or non-conforming Product, and Sena shall not be responsible for the failure of the Product to perform specified functions, or any other non- conformance caused by or attributable to: (a) any misapplication or misuse of the Product; (b) failure of Customer to adhere to any of Sena's specifications or instructions; (c) neglect of, abuse of, or accident to, the Product; or (d) any associated or complementary equipment or software not furnished by Sena.

Limited warranty service may be obtained by delivering the Product to Sena or to the international distributor it was purchased through and providing proof of purchase or receipt date. Customer agrees to insure the Product or assume the risk of loss or damage in transit, to prepay shipping charges to Sena, and to use the original shipping container or equivalent.

12.2 LIMITATION OF LIABILITY

EXCEPT AS EXPRESSLY PROVIDED HEREIN, SENA MAKES NO WARRANTY OF ANY KIND, EXPRESSED OR IMPLIED, WITH RESPECT TO ANY EQUIPMENT, PARTS OR SERVICES PROVIDED PURSUANT TO THIS AGREEMENT, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. NEITHER SENA NOR ITS DEALER SHALL BE LIABLE FOR ANY OTHER DAMAGES, INCLUDING BUT NOT LIMITED TO DIRECT, INDIRECT, INCIDENTAL, SPECIAL OR CONSEQUENTIAL DAMAGES, WHETHER IN AN ACTION IN CONTRACT OR TORT (INCLUDING NEGLIGENCE AND STRICT LIABILITY), SUCH AS, BUT NOT LIMITED TO, LOSS OF ANTICIPATED PROFITS OR BENEFITS RESULTING FROM, OR ARISING OUT OF, OR IN CONNECTION WITH THE USE OF FURNISHING OF EQUIPMENT, PARTS OR SERVICES HEREUNDER OR THE PERFORMANCE, USE OR INABILITY TO USE THE SAME, EVEN IF SENA OR ITS DEALER HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT WILL SENA OR ITS DEALERS TOTAL LIABILITY EXCEED THE PRICE PAID FOR THE PRODUCT.

12.3 HARDWARE PRODUCT WARRANTY DETAILS

WARRANTY PERIOD: Sena warranties embedded hardware Product for a period of one (1) year, and external hardware Product for a period of three (3) or five (5) years according to the Product type.

WARRANTY PROCEDURE: Upon return of the hardware Product Sena will, at its option, repair or replace Product at no additional charge, freight prepaid, except as set forth below. Repair parts and replacement Product will be furnished on an exchange basis and will be either reconditioned or new. All replaced Product and parts become the property of Sena. If Sena determines that the Product is not under warranty, it will, at the Customers option, repair the Product using current Sena standard rates for parts and labor, and return the Product at no charge in or out of warranty.

WARRANTY EXCLUSIONS: Damages caused by

- Accidents, falls, objects striking the Sena product,
- Operating the Product in environments that exceed Sena's temperature and humidity specifications,
- Power fluctuations, high voltage discharges,
- Improper grounding, incorrect cabling,
- Misuse, negligence by the customer or any other third party,
- Failure to install or operate the product (s) in accordance to their Sena User Manual,
- Failure caused by improper or inadequate maintenance by the customer or any other third party,
- Floods, lightning, earthquakes,
- Water spills,
- Replacement of parts due to normal wear and tear,
- Hardware has been altered in any way,
- Product that has been exposed to repair attempts by a third party without Sena’s written consent,
- Hardware hosting modified Sena Software, or non-Sena Software, unless modifications have been approved by Sena.
- Battery component capacity degradation due to usage, aging, and with some chemistry, lack of maintenance.

12.4 SOFTWARE PRODUCT WARRANTY DETAILS

WARRANTY PERIOD: Sena warranties software Product for a period of one (1) year.

WARRANTY COVERAGE: Sena warranty will be limited to providing a software bug fix or a software patch, at a reasonable time after the user notifies Sena of software non-conformance.

12.5 THIRD-PARTY SOFTWARE PRODUCT WARRANTY DETAILS

The warranty policy of the third-party software is conformed to the policy of the corresponding vendor.

13 Regulatory Information

13.1 FCC

FCC Rule: Part 15 Subpart C Section 15.247

FCCID: S7AIW01

FCC Compliance Statement

This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions:

(1) This device may not cause harmful interference, and

(2) This device must accept any interference received,

Including interference that may cause undesired operation

Information to User

This equipment has been tested and found to comply with limits for a Class B digital device, Pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation.

This equipment generate, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications.

However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by on or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver-Connect the equipment into an outlet a circuit different form that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

RF Exposure Statement

The equipment complies with FCC RF radiation exposure limits set forth for an uncontrolled environment. This device and its antenna must not be co-located or operation in conjunction with any other antenna or transmitter.

Do not

Any changes or modifications to the equipment not expressly approved by the party responsible for compliance could void user’s authority to operate the equipment.

13.2 CE

CE1177(!)

We, Sena Technologies, Inc., declare that the product(s):

- ProBee-ZE20S ZigBee OEM Module

to which this declaration relates is in conformity with the following standard(s) or normative document(s)

- EN 60950-1
- ETSI EN 301 489-1
- ETSI EN 301 489-17
- ETSI EN 300 328

Following the provisions of

- EMC DIRECTIVE 1999/5/EC

13.3 TELEC

Technical Regulations for Specified Radio Equipment Article 2, Section 1 (19)
Certification No:

- ProBee-ZE20SDU : 010WWBT0090
- ProBee-ZE20SDS : 010WWBT0091
- ProBee-ZE20SDC : 010WWBT0092

13.4 KCC

Type Registration

Certification No: KCC-CRM-SNA-IW01

Basic Model Number: ProBee-ZE20SDU

Series Model Number: ProBee-ZE20SDS, ProBee-ZE20SDC, ProBee-ZE20SSU, ProBee-ZE20SSC

14 RF Information

Radio Frequency Range

2.405~2.480GHz

Number of Frequency Channel

16 channels

Transmission Method

DSSS (Direct Sequence Spread Spectrum)

Modulation Method

O-QPSK (Offset Quadrate Phase Shift Keying)

Radio Output Power

+8dBm

Receiving Sensitivity

-102dBm

Power Supply

DC3.3V