



# **ZNP Host Example Library User's Guide**

Document Number: XXXXXX

**Texas Instruments, Inc.**  
San Diego, California USA

Revision	Description	Date
1.0	Initial release.	08/08/2014

## TABLE OF CONTENTS

<b>1</b>	<b>Introduction.....</b>	<b>1</b>
1.1	SCOPE.....	1
1.2	EXAMPLES .....	1
1.3	APPLICABLE DOCUMENTS.....	1
<b>2</b>	<b>Setup.....</b>	<b>2</b>
2.1	REQUIRED SOFTWARE TOOLS.....	2
2.2	SUPPORTED HARDWARE PLATFORMS .....	2
2.2.1	SMARTRF06EB BOARDS AND CC2538EM.....	2
<b>3</b>	<b>Getting Started .....</b>	<b>4</b>
3.1	BUILDING THE ZNP FIRMWARE .....	4
3.2	COMPILING AND RUNNING EXAMPLES .....	9
3.2.1	CREATING OR JOINING A NEW NETWORK .....	9
3.2.2	COMMAND LINE TRAINER .....	10
3.2.3	DATA SEND/RECEIVE .....	10
3.2.4	NETWORK TOPOLOGY .....	12
3.2.5	SERVICE DISCOVERY .....	12

# 1 Introduction

ZNP Host Example Library 1.0 is a library of examples which are built on top of a cross platform framework designed to run in a companion host target, and that will be used in combination with the embedded Z-Stack™ ZNP product.

The main focus of these examples is to show the simplicity of this framework and to make users familiar with the usage of ZNP commands.

## 1.1 Scope

This document describes how to use the ZNP Host Example Library 1.0 and discuss their theory of operation.

For guidelines on how to create your own application, please refer to *ZNP Host Developer's Guide*.

Four examples are included in the ZNP Host Example Library 1.0: cmdLineTrainer, dataSendRcv, nwkTopology, serviceDiscovery.

Unless specifically mentioned otherwise, the guidelines in this document refers to the CC2538EM, while similar approaches can be used for the other platforms.

## 1.2 Examples

Following Examples are part of the library:

a. **Command Line Trainer**

Command line application which provides all possible ZNP commands, such that the user can send any ZNP command and parameters to interactively learn the ZNP interface.

b. **Data Send/Receive (Chat application)**

Provides a command line interface which allows ZNP devices to send and receive text messages from each other.

c. **Network Topology**

Provides a description of the network topology in which the ZNP device is part of.

d. **Service Discovery**

Displays a description of the endpoints from the devices that join the network.

## 1.3 Applicable Documents

[1] ZNP Interface Specification

## 2 Setup

### 2.1 Required Software Tools

Software tools needed to evaluate sample application/s:

- a. IAR Embedded Workbench, provides a tool for compiling, linking, debugging and loading applications on the target device.

EWARM (for CC2538)

<http://www.iar.com/Products/IAR-Embedded-Workbench/ARM/>

### 2.2 Supported Hardware Platforms

#### 2.2.1 SmartRF06EB (Required only for flashing ZNP firmware) board and CC2538EM



Figure 1: SmartRF06EB with CC2538EM



Figure 2: CC2538EM

There are three ways to power SmartRF06EB; batteries, USB bus and external power supply. Power source can be selected using the power source selection switch (S502), seen below in **Figure 3**. Main power supply switch (S501) cuts power to the SmartRF06EB. **For Flashing and Debugging:** Make sure USB cable is connected to the PC.



Figure 3 S501 and S502

## 3 Getting Started

### 3.1 Building the ZNP Firmware

- Make sure all development software tools have been installed
- Consult section 2.2 to Power up the board.
- If Windows prompts to install a device driver, don't let it connect to Windows Update. Instead, let Windows try to find the required driver automatically. If that fails, browse to: *C:\Program Files\IAR Systems\<Embedded Workbench>\<arm/8051>\drivers\Texas Instruments* to locate the necessary files.
- Select IAR project workspace  
(C:\Texas Instruments\Z-Stack Home 1.2.0\Projects\zstack\ZNP\CC2538 ZNP.eww),  
and open it with IAR. Select cc2538 configuration from the *Workspace* pull-down menu.

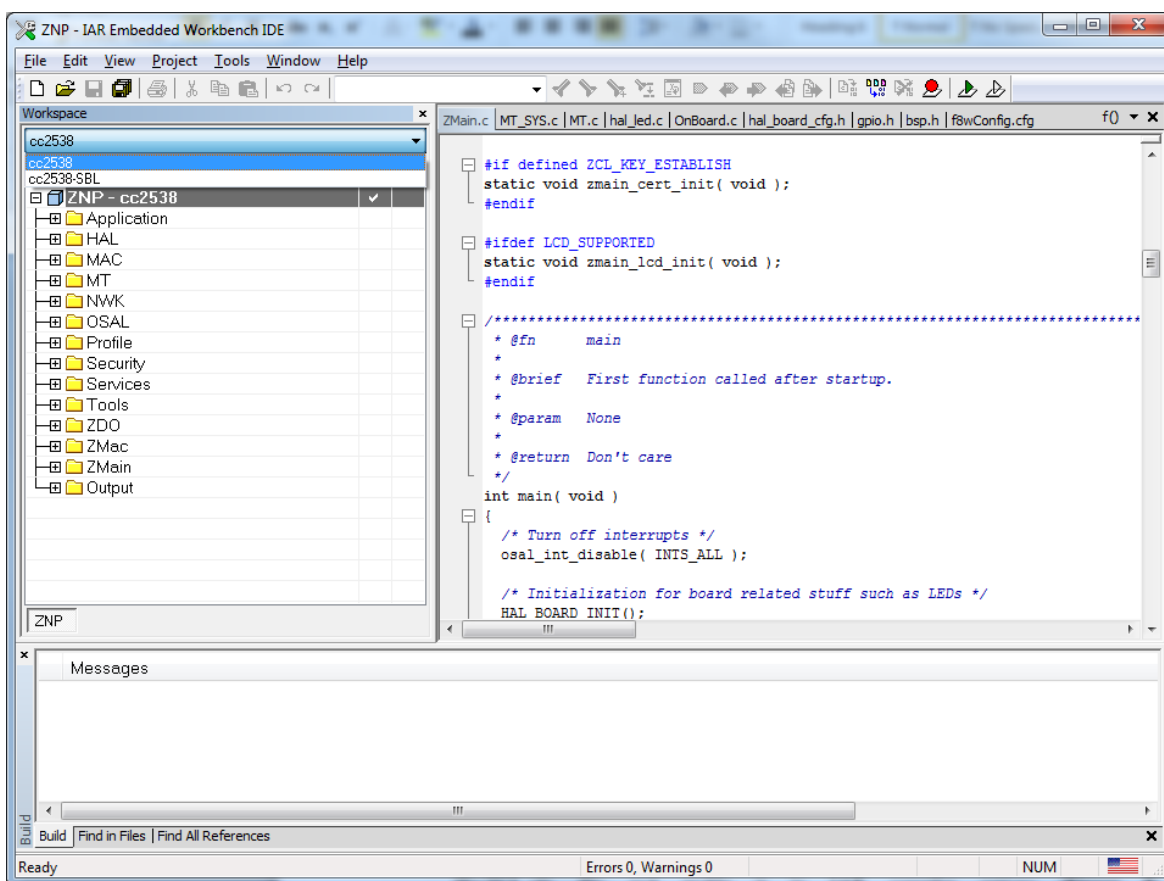


Figure 4: Selecting Correct Configuration

- Select the pull down menu *Project* and click on *Options*. Then select *C/C++ Compiler* and go to the *Preprocessor* tab and add the following symbols if they are not already there.

HAL\_SPI=TRUE

MT\_UART\_DEFAULT\_OVERFLOW=FALSE

HAL\_UART\_USB

USB\_SETUP\_MAX\_NUMBER\_OF\_INTERFACES=5

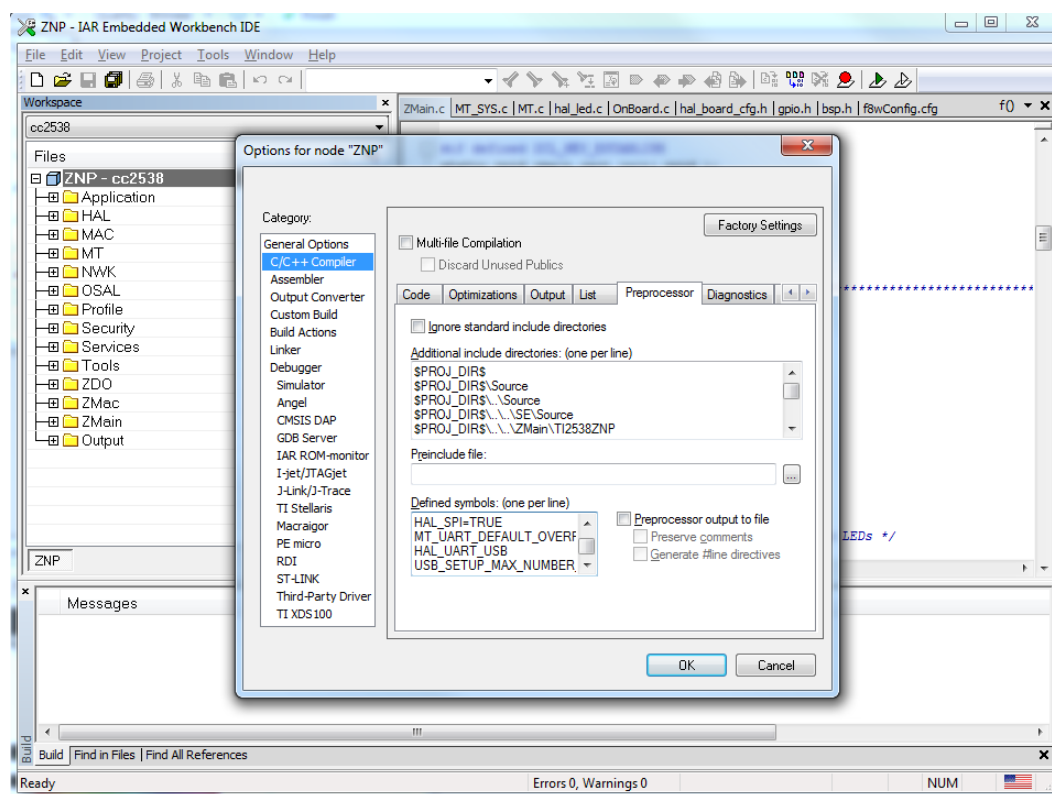


Figure 5: Defining symbols in Preprocessor



- Build the application by pulling down the *Project* menu and clicking on **Rebuild All**:

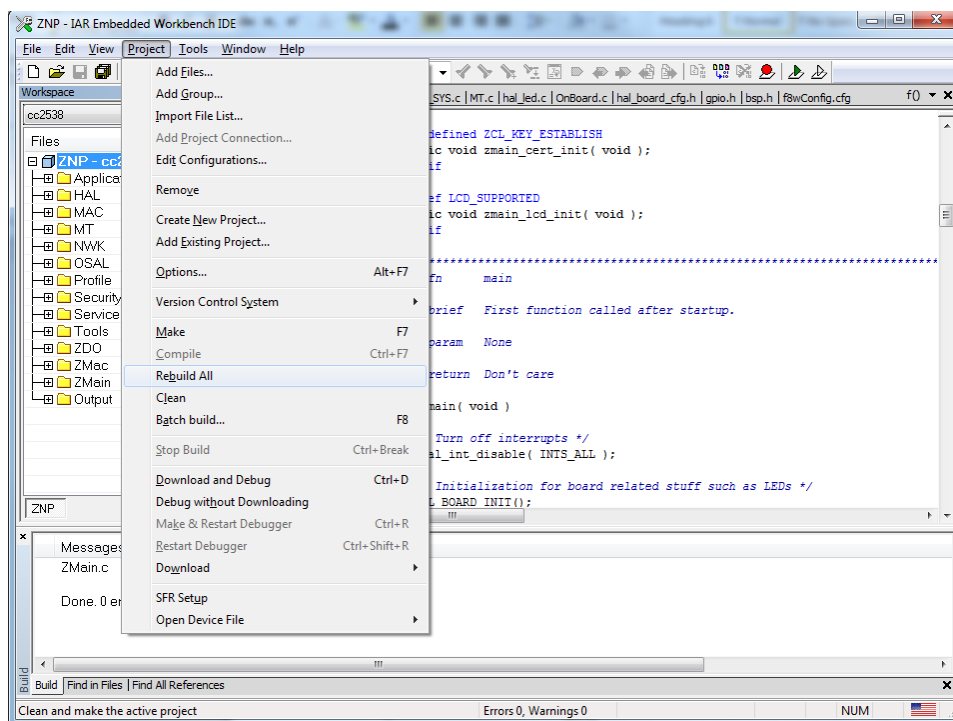


Figure 6: Building ZNP Firmware

- Erase flash prior to programming new image:

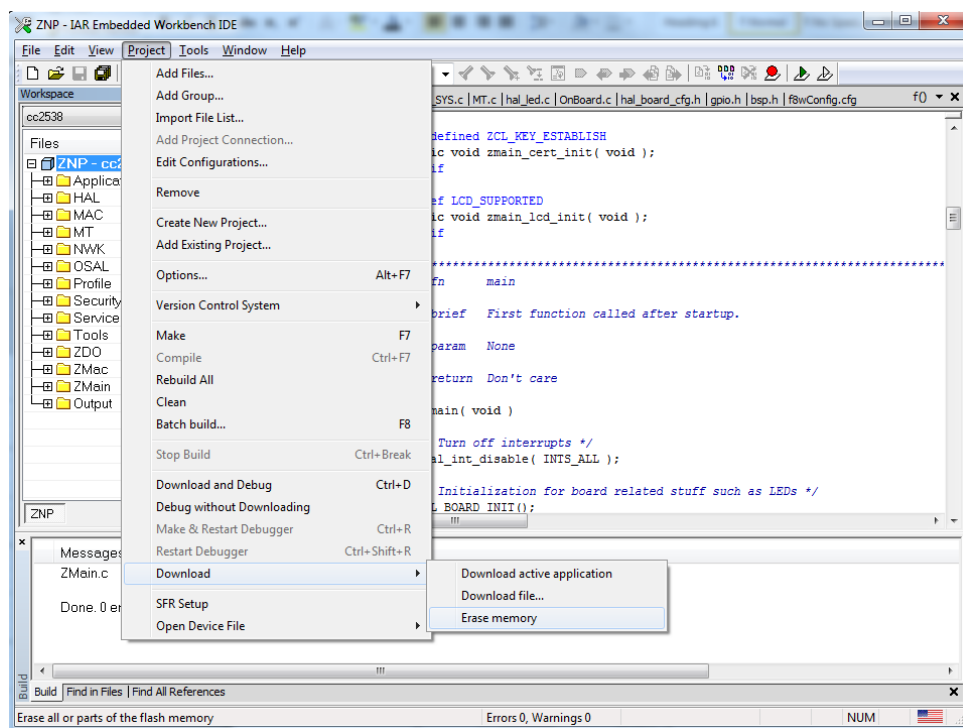


Figure 7: Erase Memory in CC2538

- Program (download) the ZNP firmware into the module by pulling down the *Project* menu and clicking on **Download and Debug**:

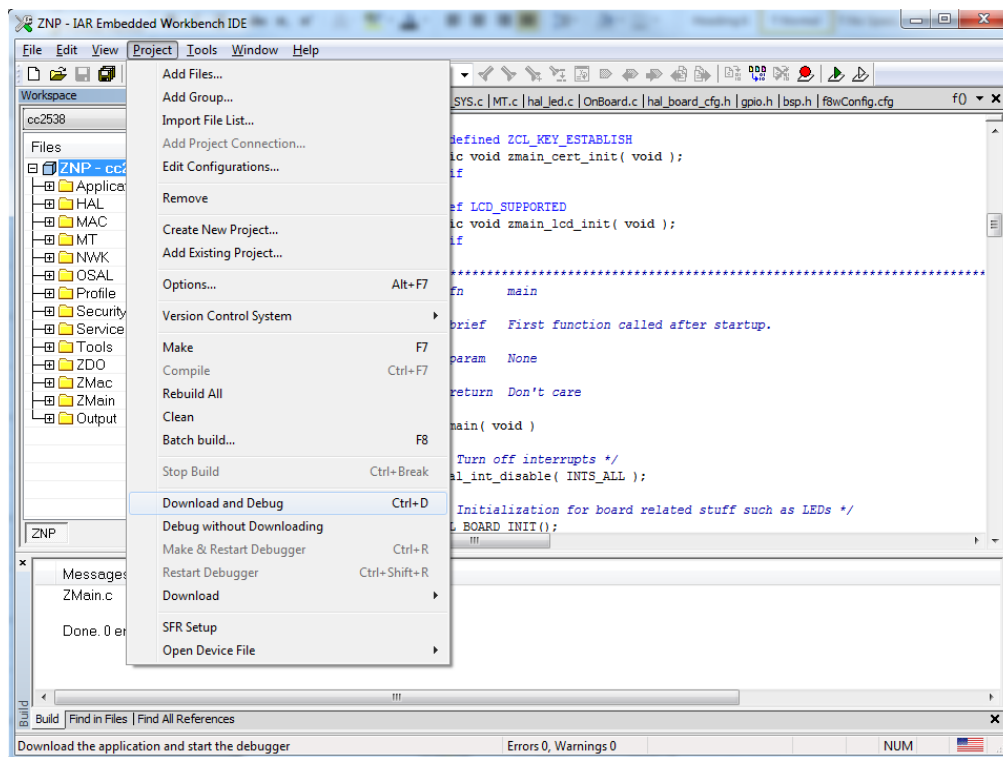


Figure 8: Downloading Sample Application

- After downloading to the CC2538EM is complete, exit the debugging mode by pulling down the *Debug* menu and clicking on **Stop Debugging**:

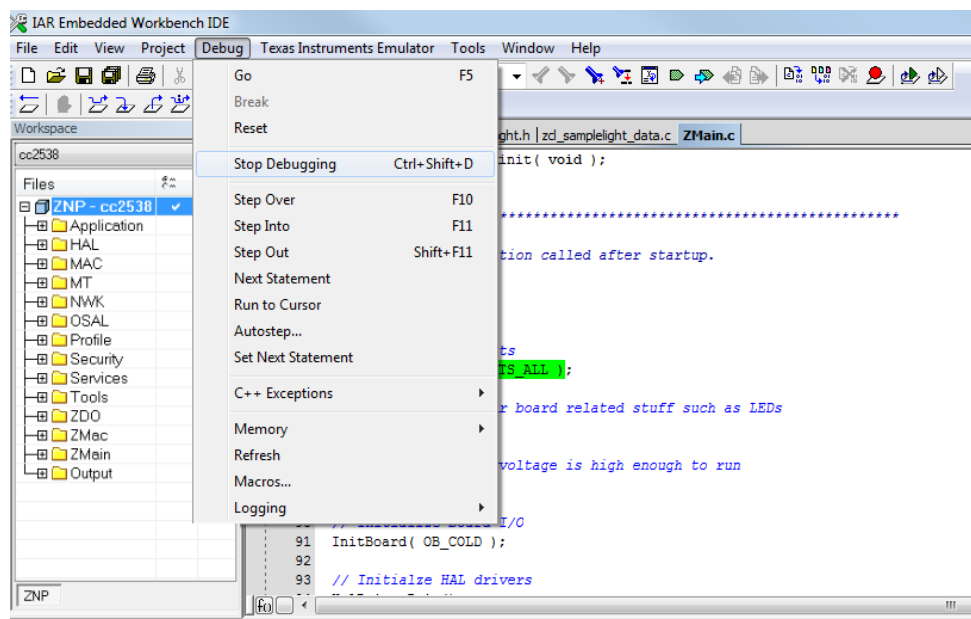


Figure 9: Stop Debugging

- The cc2538EM must be detached from the SmartRF06 board after the firmware image is flashed.
- If the cc2538EM will be connected to the host through USB then you will need to reposition the jumper link to the position shown in red in the picture below.



**Figure 10: Jumper position for USB host connection**

## 3.2 Compiling and Running Examples

After flashing the CC2538EM with the ZNP firmware, the examples will need to be compiled.

To compile the examples, open a terminal window go to the directory where the example is located i.e. “/ZNP\_Host/examples/dataSendRcv/build/gnu” if you are in a Linux platform. While you are in this directory type **make** to compile the example.

If the example has been already compiled then in the same directory enter the following command.

**./exampleName.bin /dev/ttyACM**X****

exampleName is the name of the example you want to run, i.e. dataSendRcv.bin. The **X** is a number, for example, you would use ttyACM1 if your ZNP device is associated to that port.

### 3.2.1 Creating or Joining a new Network

Whenever you run any of the examples in the library, you will be prompted if you will like to start a new network. If your ZNP device has not joined a network or you want to start a new network then type **y** and press enter.

```
Do you want to start/join a new network? (y/n)
y
```

Starting a new network will reset the configuration in the ZNP. You will be then prompted to select what type of device you want your ZNP to be, coordinator, router, or end device.

```
Resetting ZNP
ZNP Version: 2.6.1
Enter device type c: Coordinator, r: Router, e: End Device:
c
```

After this, you will need to enter the channel in which you want your device to operate in, and the device will start or join a new network.

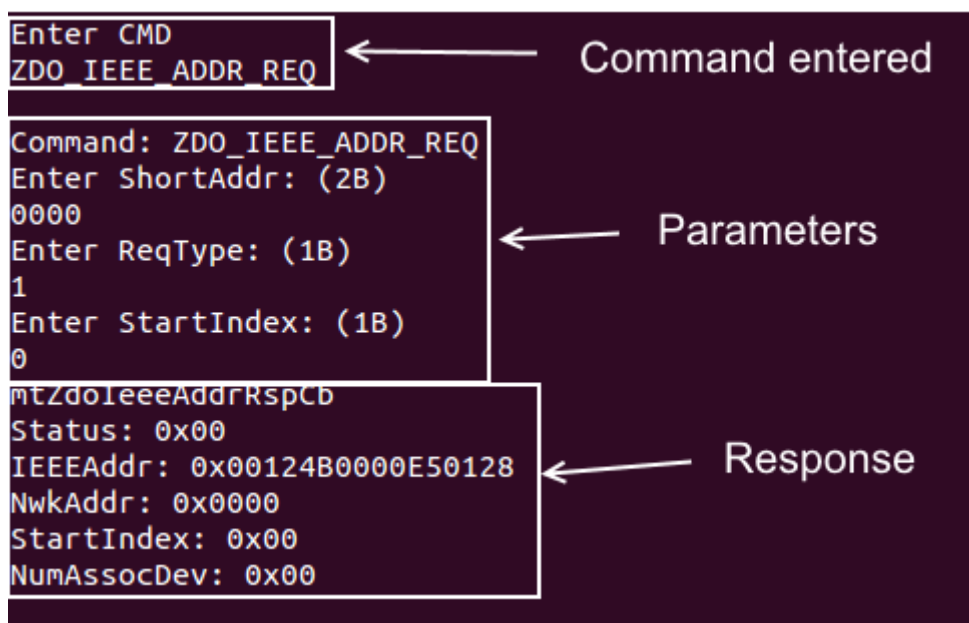
```
Enter channel 11-26:
19

EndPoint: 1
Network Starting
Network Started
Network up
```

### 3.2.2 Command Line Trainer

After the network has been set up, you will be able to enter the commands that you wish to send. Some of the features of this example are:

- Press the Tab key twice to see all the available commands.
- Press Tab once to autocomplete the command that you are entering.
- Press Enter to display any incoming messages in queue.
- Use the up and down arrow keys to see the history of commands previously entered.
- After typing the full command name, you can press tab twice and a description of the command will be displayed.
- Press enter after typing the command name to select the command and fill out the values for the parameters.



The screenshot shows a terminal window with a dark background and light-colored text. It illustrates the process of entering a command and receiving a response. Three white boxes with arrows point to specific parts of the output:

- Command entered:** Points to the first box containing the command `ZDO_IEEE_ADDR_REQ`.
- Parameters:** Points to the second box containing the command details: `Command: ZDO_IEEE_ADDR_REQ`, `Enter ShortAddr: (2B)` with value `0000`, `Enter ReqType: (1B)` with value `1`, and `Enter StartIndex: (1B)` with value `0`.
- Response:** Points to the third box containing the response details: `mtZdoIeeeAddrRspCb`, `Status: 0x00`, `IEEEAddr: 0x00124B0000E50128`, `NwkAddr: 0x0000`, `StartIndex: 0x00`, and `NumAssocDev: 0x00`.

### 3.2.3 Data Send/Receive

After the network is set up, if you initialized the device to be a coordinator, the application will wait for a device to join. Then when a device joins the network, it will display the information of the new device.

```
Waiting for device to join the network...
New device joined network.
NwkAddr: 0xBF1D
Number of Endpoints: 1
Active Endpoints: 1
```

Before you can start sending messages, you will see a list of available addresses and endpoints to send messages to.

```
Available addresses:
Type: COORDINATOR
NwkAddr: 0x0000
Number of Endpoints: 1
Active Endpoints: 1

Type: ROUTER
NwkAddr: 0xBF1D
Number of Endpoints: 1
Active Endpoints: 1
```

Fill in the destination address and the destination endpoint for the device that you want to exchange messages with.

```
Enter DstAddr:
BF1D
Enter DstEndpoint:
1
```

Then you will be able to send and receive messages with the selected device. To change the destination of the messages type CHANGE, this will allow you to enter a different destination address and endpoint.

```
Enter message to send or type CHANGE to change the destination:
Hello World!

Message transmited Succesfully!!
```

### 3.2.4 Network Topology

After the network is set up, press enter whenever you want the network topology to be displayed.

```
Press Enter to discover Network Topology:

Node Address: 0x0000   Type: COORDINATOR
Children: 2
    Address: 0x2062   Type: ROUTER
    Address: 0xD6A8   Type: ROUTER

Node Address: 0x2062   Type: ROUTER
Children: 0

Node Address: 0xD6A8   Type: ROUTER
Children: 1
    Address: 0x484B   Type: END DEVICE
```

### 3.2.5 Service Discovery

Wait for a device to join the network and the description of the new device will be displayed automatically.

```
New device joined network.
NwkAddr: 0x631D
Number of Endpoints: 1
    Endpoint: 0x03
    ProfileID: 0x0104
    DeviceID: 0x0100
    DeviceVersion: 0x00
    NumInClusters: 5
        InClusterList[0]: 0x0000
        InClusterList[1]: 0x0003
        InClusterList[2]: 0x0004
        InClusterList[3]: 0x0005
        InClusterList[4]: 0x0006
    NumOutClusters: 1
        OutClusterList[0]: 0x0000
```