TEXAS INSTRUMENTS

Z-Stack Home Sample Application User's Guide

Document Number: SWRU354

Texas Instruments, Inc.
San Diego, California USA

Revision	Description	Date
1.0	Initial release.	06/12/2013
1.1	Added Sample Applications after HA 1.2 implementation. Added clarification of help display.	11/25/2013
1.2	Cleanup for Z-Stack Home 1.2.1 release.	07/03/2014
1.3	Added CC2630/CC2650 and SensorTag sample application description.	02/20/2015

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

ZigBee Home Automation (ZHA) is one of the public application profiles released for the ZigBee 2012 specification. HA networks scale from 2 to 500 nodes, and are easily installed by either novice users, or home automation professionals. Installation concepts are simple and uniform across multiple OEM vendors.

ZigBee Home Automation is primarily focused on sporadic real time control of devices, that is, the network is normally quiet, but when a user presses a button on a device, he expects to see the result of that button press across the network quickly.

More information can be found in www.zigbee.org, and in the official ZigBee Home Automation Public Application Profile document by the ZigBee alliance (document 05-3520).

The Home Automation (HA) Sample application (part of the Z-Stack Home installer) is the optimal starting point to build your own HA application on top of Texas Instruments' Z-Stack (www.ti.com/z-stack).

1.1 Scope

This document describes how to use Z-Stack Home Sample Applications and discuss their theory of operation. For a thorough description of Home Automation, the reader shall refer to the ZigBee Home Automation specification [1] available from www.zigbee.org. For an in-depth description of the Z-Stack in general and the Z-Stack Home Sample Applications in particular, please refer to the *Z-Stack Developer's Guide* [3] and the *Z-Stack Home Developer's Guide* [2], respectively. These documents are available as part of the Z-Stack Home package, and provide a good reference, e.g. for changing Z-Stack configuration parameters such as Channel and the network PAN ID to use.

For guidelines on how to create your own application based on the included Sample Applications, please also refer to *Z-Stack Home Developer's Guide*.

Seven sample applications are included in the Z-Stack Home installer: SampleDoorLock, SampleDoorLockController, SampleLight, SampleSwitch, SampleHeatingCoolingUnit, SampleTemperatureSensor, SampleThermostat, and the SensorTag sample application supported only on the CC2650 SensorTag. Each of them supports Coordinator, Router and End Device configurations when used with CC253x platforms and in CC2630/CC2650 platforms End Device is the only configuration supported. The Sample Switch also contains multiple build configurations that support OTA. For further details regarding using OTA, please refer to *Z-Stack OTA Upgrade User's Guide* [4].

Unless specifically mentioned otherwise, the guidelines in this document refer to the CC2538 SoC, while similar approaches can be used for the other platforms.

1.2 Sample Projects

The Following sample projects are supported:

a. **SampleLight / SampleSwitch**

   SampleLight: A light that can be turned on/off locally or remotely.

   SampleSwitch: A switch that acts as a remote to turn a light on/off .

b. **SampleDoorLock** / **SampleDoorLockControlle**r

   SampleDoorLock: A door lock that can be locked/unlocked locally or remotely with ability to change master PIN.

   SampleDoorLockController: A controller that locks/unlocks the door lock device based on user PIN input.

c. **SampleTemperatureSensor / SampleThermostat / SampleHeatingCoolingUni**t

   SampleTemperatureSensor: Sends current temperature reading to the thermostat.

   SampleThermostat: A unit that receives temperature information from the Temperature Sensor and adjusts the room temperature by sending heating/cooling commands to the Heating/Cooling Unit.

   SampleHeatingCoolingUnit: A unit that heats or cools based on the received information from the thermostat.

Sample projects supported by CC2630/CC2650 platforms:

- SampleSwitch
- SampleDoorLock / SampleDoorLockController
- SampleTemperatureSensor
- SensorTag: Combination of a light switch and temperature sensor in a single application.

1.3 Definitions, Abbreviations, Acronyms

Term	**Definition**
BB	Battery Board
HVAC	Heating, Ventilation, and Air Conditioning
PIN	Personal Identification Number
LED	Light Emitting Diode
LCD	Liquid Crystal Display
ZC	ZigBee Coordinator
ZCL	ZigBee Cluster Library
ZED	ZigBee End Device
ZHA	ZigBee Home Automation
ZR	ZigBee Router

1.4 Applicable Documents

[1] ZigBee document 05-3520-29 ZigBee Home Automation Specification

[2] Texas Instruments document SWRA441, Z-Stack Home Developer’s Guide

[3] Texas Instruments document SWRA176, Z-Stack Developer's Guide.

[4] Texas Instruments document SWRA353, Z-Stack OTA Upgrade User’s Guide.

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.

   EW8051 (for CC2530)

   http://www.iar.com/Products/IAR-Embedded-Workbench/8051/

   EWARM (for CC2538 and CC2630/CC2650)

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

b. SmartRF Flash Programmer 2 Tool - for SmartRF06EB+CC2538EM and or SmartRF Flash Programmer Tool - for SmartRF05EB+CC2530EM. These can be found here:

   http://www.ti.com/tool/flash-programmer

c. SmartRF Studio. It includes Ubiqua USB dongle driver and other necessary software www.ti.com/smartrfstudio.

d. Ubiqua Protocol Analyzer from Ubilogix (*www.ubilogix.com*) or other type of network analyzer that supports ZigBee Home Automation profile decoding. Ubiqua requires using the CC2531 USB Dongle to capture the network traffic.

e. The CC-Debugger, http://www.ti.com/tool/cc-debugger, is a small programmer and debugger for the TI Low Power RF System-on-Chips. This is required for use programming and debugging using the Zlight2 platform.

2.2 Supported Hardware Platforms

2.2.1 SmartRF06EB boards with CC2538EM or CC2630/CC2650EM

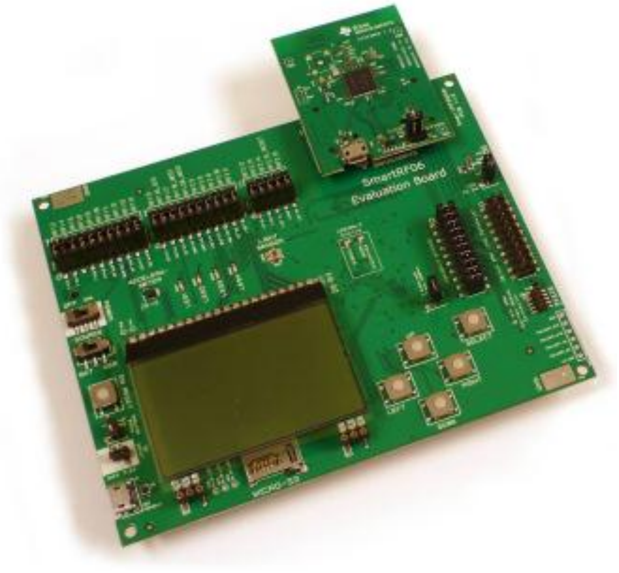


Figure 1: SmartRF06EB with CC2538EM

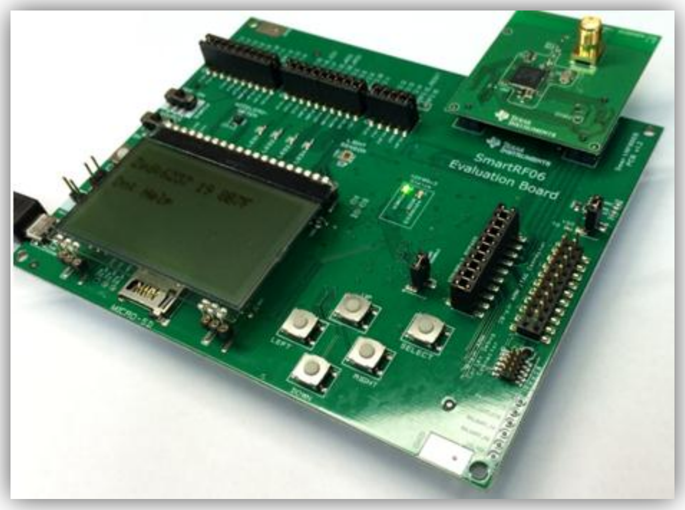


Figure 2 SmartRF06 with CC2650EM

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. Consult **Table 1** and **Figure 4** for Button Positions:

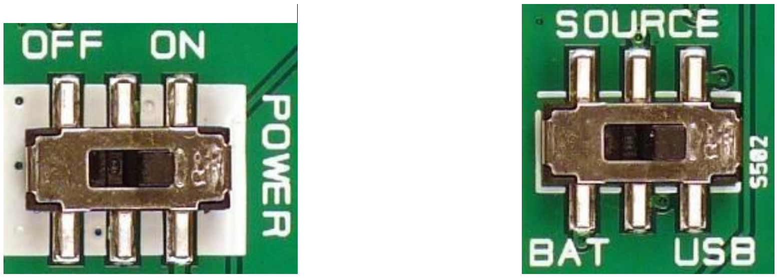


Figure 3: S501 and S502

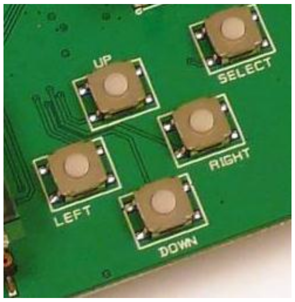


Figure 4: Button Positions

	Button
SW1	UP
SW2	RIGHT
SW3	DOWN
SW4	LEFT
SW5	SELECT

Table 1: SmartRF06 Key Positions

2.2.2 SmartRF05EB/BB boards with CC2530EM

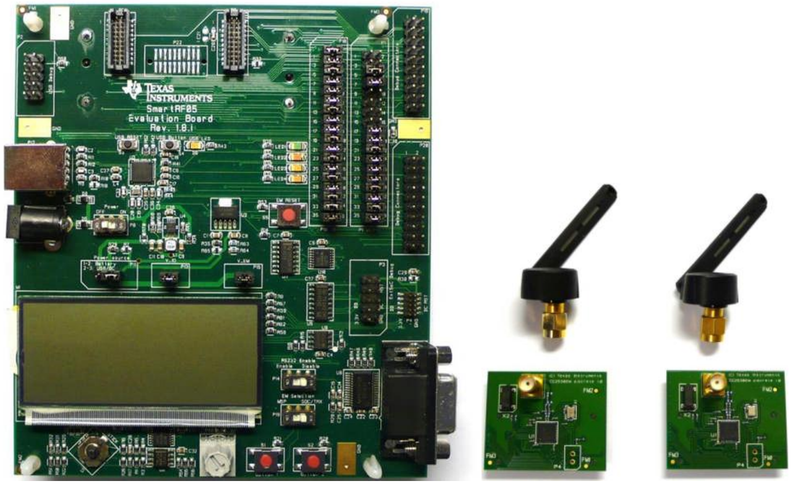


Figure 5: SmartRF05 Evaluation Board, Two CC2530EM Evaluation Modules and Antennas

There are 3 ways to supply power to the SmartRF05EB board: batteries, USB connection, or a DC power supply. To provide power from batteries, pins 1-2 of jumper block P11 must be connected. Otherwise, connect pins 2-3 to use USB or DC. Board can be powered ON or OFF using switch P8. **For Flashing and Debugging**: Make sure USB cable is connected to the PC.

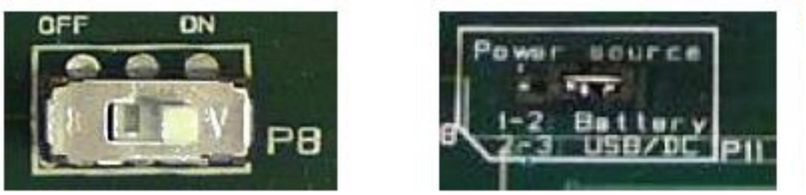


Figure 6: Power Switch and Power Source Selection

In general, first line of the LCD display shows network information of the device and the second line shows the EZ-Mode pairing status.

When pairing devices using EZ-Mode, a device can either be an “initiator” or a “target”. An initiator will display target’s network information on the second line of the LCD screen once pairing is successful. Initiator sends information to the target device.

Joystick (**Figure 7**) and buttons (**Figure 8**) on the SmartRF05 board are used by all sample applications.

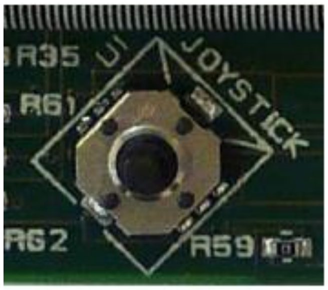


Figure 7: Joystick on SmartRF05EB REV 1.8.1

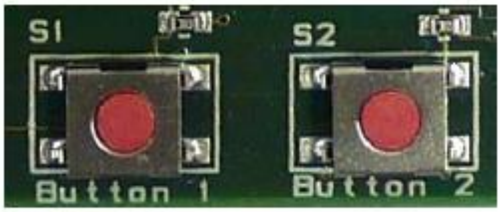


Figure 8: Buttons on SmartRF05EB REV 1.8.1

Joystick multi-position switch (5 positions) is used in the sample applications. For the sample applications, these "positions" are titled "Switches" to distinguish them from the Buttons.

	Joystick Position
SW1	UP
SW2	RIGHT
SW3	DOWN
SW4	LEFT
SW5	CENTER PRESSED

Table 2: SmartRF05 Key Positions

Above table shows various key positions. "Button 1" functions as a "shift" key. **To restore the device to factory settings, press shift (Button 1) + SW5**. This function works regardless of device mode

2.2.3 Zlight2 with CC2531

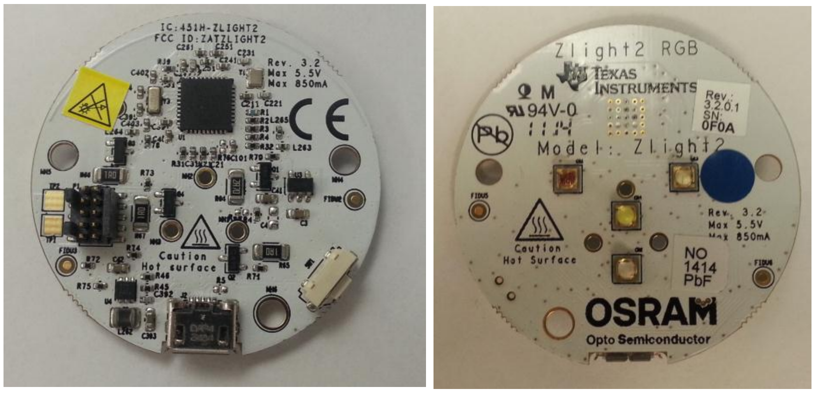


Figure 9: CC2531Zlight2

The Zlight2 platform is supported as an alternative platform (build configuration) in the Sample Light workspace for CC2530.

The Zlight2 is powered via the USB connector. Note, the USB connector ONLY provides power and cannot be used for programming or debugging.

For Flashing and Debugging: Make sure the USB cable is connected to a powered USB host (i.e. PC) and the CC debugger is connected to the 10 pin header as shown in the following figure.

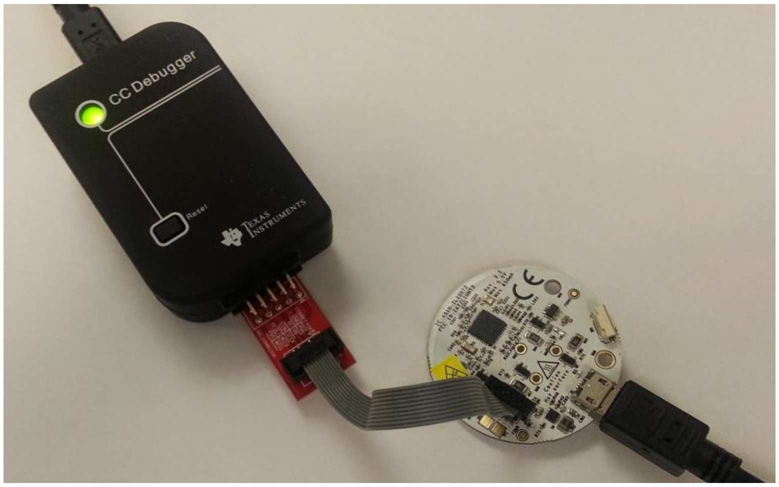


Figure 10: USB and CC Debugger connected

The Zlight2 has only one button, shown in the following figure. When pressed, this button will force the Sample Light to leave any network it is connected to and perform a system reset. This button is mapped to SW2.

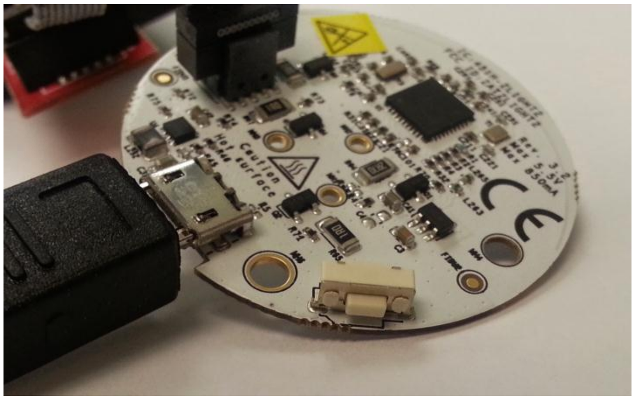


Figure 11: Zlight2 Button

2.2.4 CC2650 SensorTag

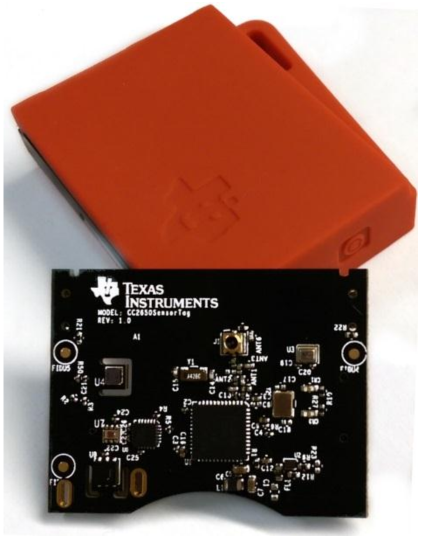


Figure 12 CC2650 SensorTag

The CC2650 SensorTag is a small and portable development platform capable of running the ZigBee/6LoWPAN stack as well as BLE stack. This platform contains 9 low power sensors.

- Humidity/Temperature
- Object Temperature
- Ambient Light
- Pressure
- Accelerometer
- Gyro
- Magnetometer
- Digital Microphone
- Magnet sensor

Currently the features supported in the SensorTag sample application are temperature sensor and a light switch.

To Flash and debug the SensorTag it needs to be connected to the SmartRF06 board as shown in Figure 13. There is no need for external power source for the SensorTag, the SmartRF06 board will supply power through the debugger cable.

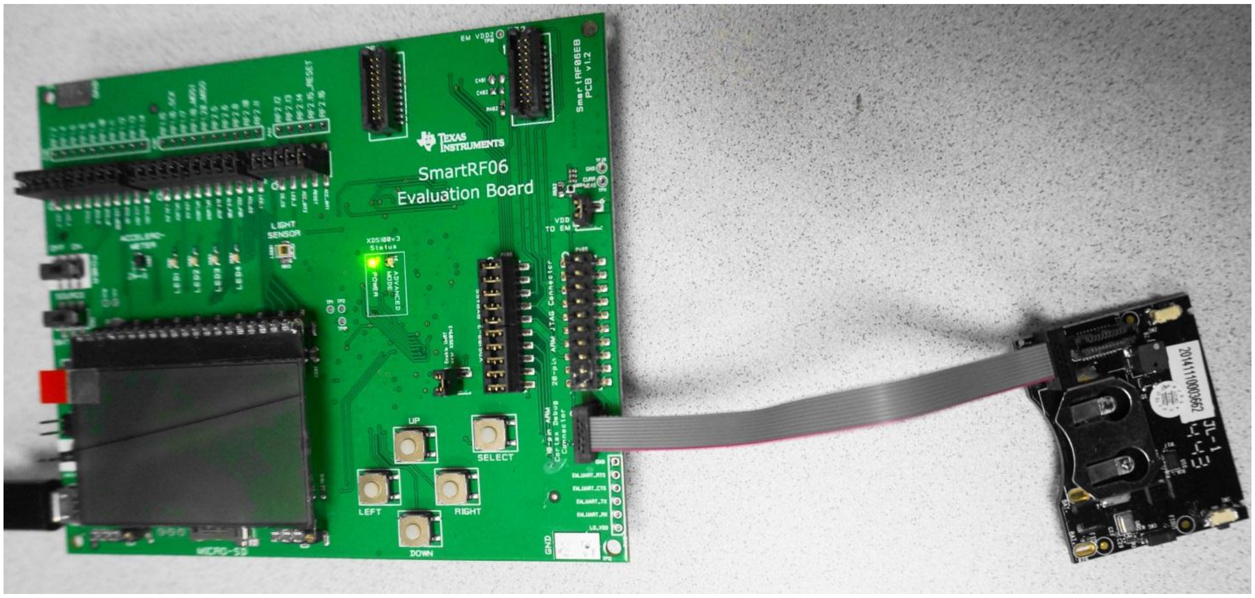


Figure 13 Debugger connection for SensorTag

2.3 ZigBee Cluster Libraries Required for Sample Applications

Sample applications use various clusters. Table 3 identifies clusters common to all sample applications, and table 4 identifies clusters that are unique and mandatory to particular sample applications. Sample applications include all mandatory features/clusters to achieve ZigBee certification for that device.

For proper operation, the following clusters are enabled in all sample applications. These are already set in the IAR Workbench preprocessor compile flags for each application:

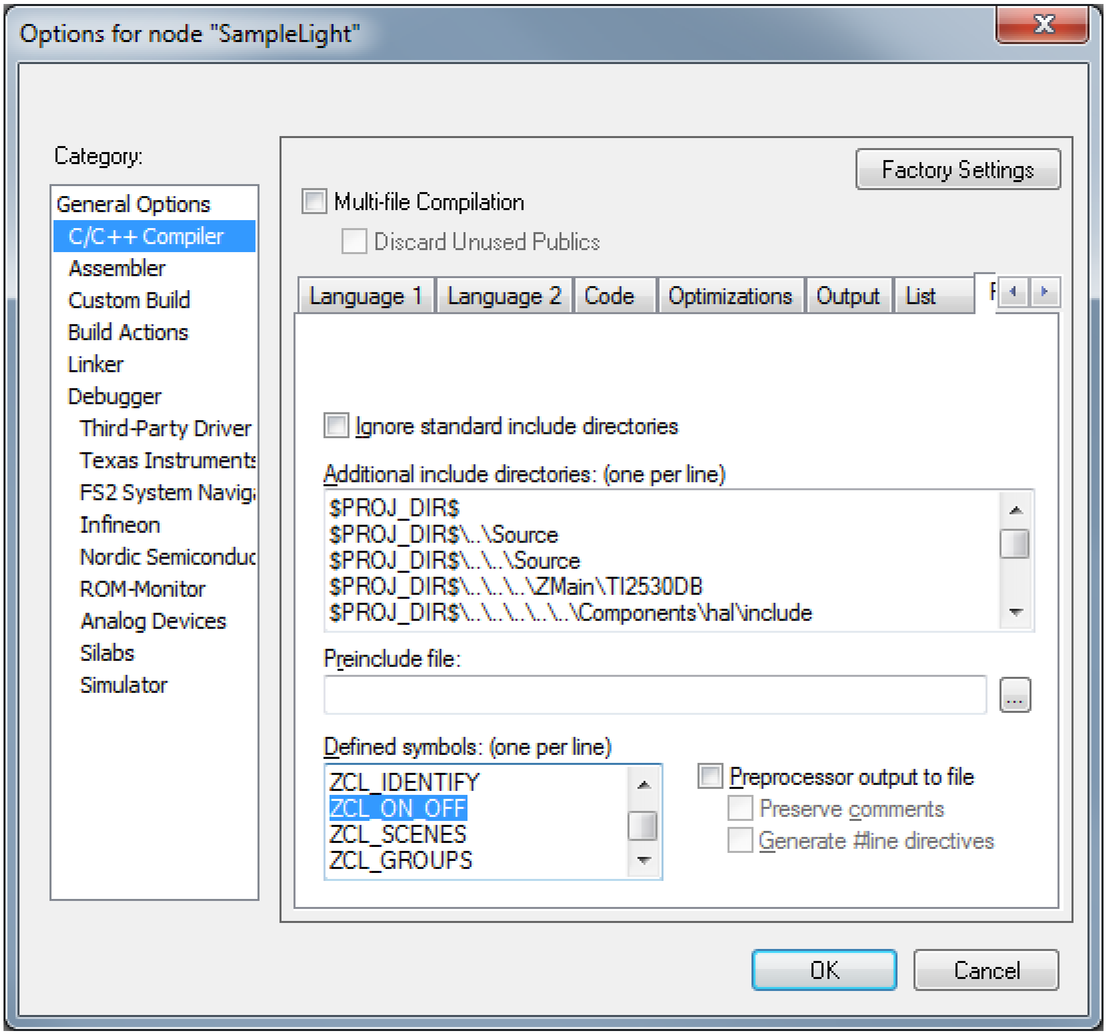
-DZCL_READ
-DZCL_WRITE
-DZCL_REPORT
-DZCL_EZMODE
-DZCL_BASIC
-DZCL_IDENTIFY

Table 3: Clusters required across all Sample Applications

Following table lists other mandatory clusters for each sample application. User may enable specific clusters in IAR options (**Figure 14)**. Full list is defined in f8wZCL.cfg.

Application	Mandatory Cluster
SampleLight	On/Off, Scenes, Groups
SampleSwitch	On/Off
SampleDoorLock	DoorLock, Scenes, Groups
SampleDoorLockController	DoorLock, Scenes, Groups
SampleThermostat	On/Off, Thermostat
SampleHeatingCoolingUnit	On/Off, Thermostat
SampleTemperatureSensor	Temperature Measurement
SensorTag	On/Off, Temperature Measurement

Table 4: Supported Mandatory Clusters



□ Multi-file Compilation

□ Discard Unused Publics

□ Ignore standard include directories

Additional include directories: (one per line)

```
$PROJ_DIR$
$PROJ_DIR$\..\Source
$PROJ_DIR$\..\..\Source
$PROJ_DIR$\..\..\..\ZMain\TI2530DB
$PROJ_DIR$\..\..\..\..\..\Components\hal\include
```

Preinclude file:

Defined symbols: (one per line)

```
ZCL_IDENTIFY
ZCL_ON_OFF
ZCL_SCENES
ZCL_GROUPS
```

□ Preprocessor output to file

□ Preserve comments

□ Generate #line directives

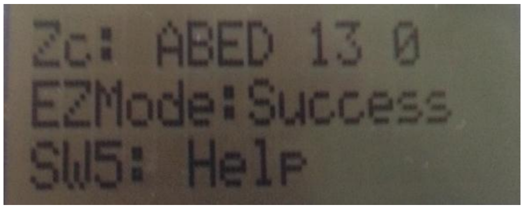
Figure 14: IAR Project Compile option to enable clusters

2.4 Sample Application User Interfaces

Sample applications have a common main and help screen modes. After the device is on the network, it will display the main mode screen as shown in Figure below. In the main mode, the user can view the network information, EZ-Mode status, and the status of local functions and responses from the local and remote devices. The help mode outlines the functionality of the joystick buttons from the main/help modes.

Note: Only three lines of display are supported on the LCD, so not all joystick/button mappings are displayed in the help mode. Please refer to the following sections for a complete mapping of button functions for each sample application.

Device type - PAN ID - Channel ID - Short Address =>
Status/Command Updates =>
Switch Help =>



```
Zc: ABED 13 0
EZMode:Success
SW5: Help
```

Figure 15: Main Mode Screen Example

ZigBee Coordinator device automatically forms the network. To add other ZigBee devices (ZigBee Routers and ZigBee End Devices), EZ-Mode must be used (SW2 on the joystick).

2.4.1 SampleLight, SampleSwitch

SampleLight (ZC)

MAIN MODE / HELP MODE		
Joystick/Button	Functionality	Comment
SW1	Toggle local light	Toggle On/Off
SW2	Invoke EZ-Mode	
SW4	Toggle permit join state	Not necessary for EZ-Mode
SW5	Go into help mode	Describes the function of the buttons in main mode

Table 5: SampleLight Functionality Mapping

SampleSwitch (ZED)

MAIN MODE / HELP MODE		
Joystick/Button	Functionality	Comment
SW1	Toggle remote light	Toggle On/Off
SW2	Invoke EZ-Mode	
SW5	Go into help mode	Describes the function of the buttons in main mode

Table 6: SampleSwitch Functionality Mapping

2.4.2 SampleDoorLock, SampleDoorLockController

SampleDoorLock (ZC)

MAIN MODE / HELP MODE		
Joystick/Button	Functionality	Comment
SW1	Toggle local door lock	Toggle Lock/Unlock
SW2	Invoke EZ-Mode	
SW3	Go into "*change master PIN*" mode	Change Master PIN. This pin should match on door lock and door lock controller. Details in Table below
SW4	Toggle permit join state	Not necessary for EZ-Mode
SW5	Go into help mode	Describes the function of the buttons in main mode

CHANGE MASTER PIN MODE		
Joystick/Button	Functionality	Comment
SW1	Increase PIN number	Increases selected digit.
SW3	Decrease PIN number	Decreases selected digit
SW5	Enter PIN number	Pressed to confirm each digit of master pin

Table 7: SampleDoorLock Functionality Mapping

SampleDoorLockController (ZED)

MAIN MODE / HELP MODE		
Joystick/Button	Functionality	Comment
SW1	Toggle remote door lock	Sends PIN to door lock to lock/unlock the door
SW2	Invoke EZ-Mode	
SW3	Go into "change PIN" mode	Change Master PIN. This pin should match on door lock and door lock controller. Details in Table below
SW5	Go into help mode	Describes the function of the buttons in main mode

CHANGE PIN MODE		
Joystick/Button	Functionality	Comment
SW1	Increase PIN number	Increases selected digit.
SW3	Decrease PIN number	Decreases selected digit
SW5	Enter PIN number	Pressed to confirm each digit of master pin

Table 8: SampleDoorLockController Functionality Mapping

2.4.3 SampleThermostat, SampleTemperatureSensor, SampleHeatingCoolingUnit

SampleThermostat (ZC)

MAIN MODE / HELP MODE		
Joystick/Button	Functionality	Comment
SW1	Sets heating setpoint	Enters Heating Setpoint mode to change preset temperature when heating unit turns on
SW2	Invoke EZ-Mode	
SW3	Set cooling setpoint	Enters Cooling Setpoint mode to change preset temperature when cooling unit turns on
SW4	Toggle permit join state	Not necessary for EZ-Mode
SW5	Go into help mode	Describes the function of the buttons in main mode

HEATING/COOLING SETPOINT MODE		
Joystick/Button	Functionality	Comment
SW1	Increase temperature	Increases set Temperature
SW3	Decrease temperature	Decreases set Temperature
SW5	Save temperature Setting	Saves the temperature for that setpoint

Table 9: SampleThermostat Functionality Mapping

SampleHeatingCoolingUnit (ZR or ZED)

MAIN MODE / HELP MODE		
Joystick/Button	Functionality	Comment
SW2	Invoke EZ-Mode	
SW5	Go into help mode	Describes the function of the buttons in main mode

Table 10: SampleHeatingCoolingUnit Functionality Mapping

SampleTemperatureSensor (ZED or ZR)

MAIN MODE / HELP MODE		
Joystick/Button	Functionality	Comment
SW1	Increase temperature	Reported Temperature is simulated. Use this switch to increase the reported temperature.
SW2	Invoke EZ-Mode	
SW3	Decrease temperature	Reported Temperature is simulated. Use this switch to decrease the reported temperature.
SW4	Toggle permit join state	Not necessary for EZ-Mode
SW5	Go into ‘help’ mode	Describes the function of the buttons in main mode

Table 11: SampleTemperatureSensor Functionality Mapping

2.4.4 SensorTag

SensorTag(ZED)

MAIN MODE / HELP MODE		
Button	Functionality	Comment
SW1	Invoke EZ-Mode	Invokes EZ-Mode for the switch functionality
SW2	Toggle remote light	Toggle On/Off
SW1 + SW2	Invoke EZ-Mode for temperature sensor	This key combination will invoke EZ-Mode for the temperature sensor.

Table 12: SensorTag Functionality Mapping

3 Getting Started

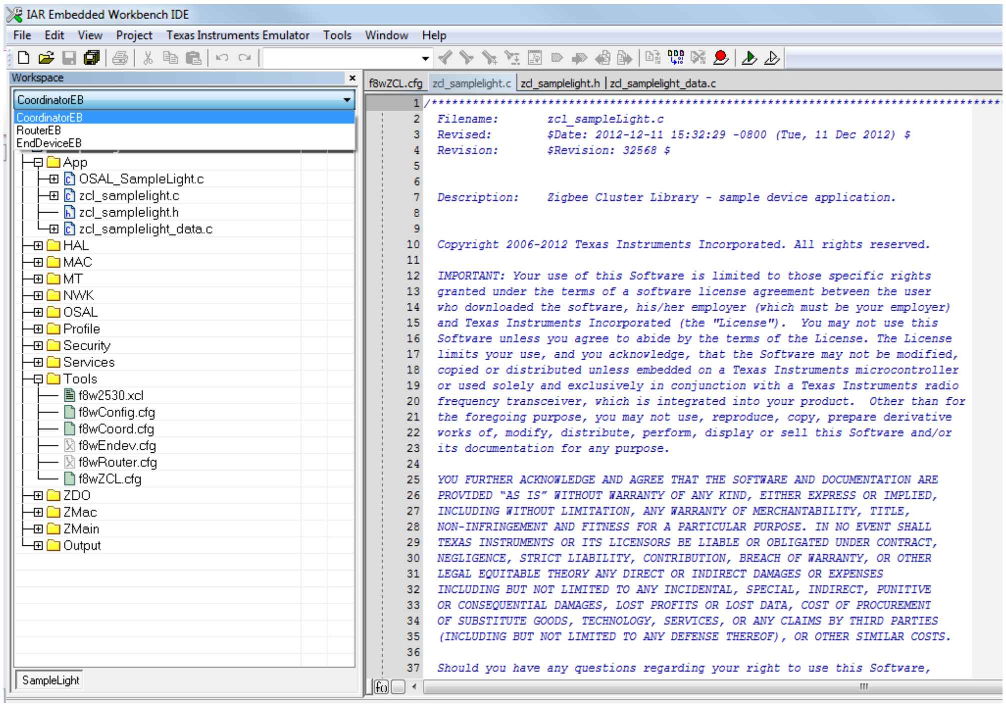
3.1 Building the Sample Applications

All Sample applications configured as end devices/routers are compiled with option “HOLD_AUTO_START” and do not join the network automatically. Devices join the network when EZMODE is initiated. The process of building a sample application is the same for any application. The only differences are whether the device is a coordinator, router, or end device, and which ZCL clusters to include.

3.1.1 CC2530/CC2538

This step-by-step process will use the SampleLight application as an example.

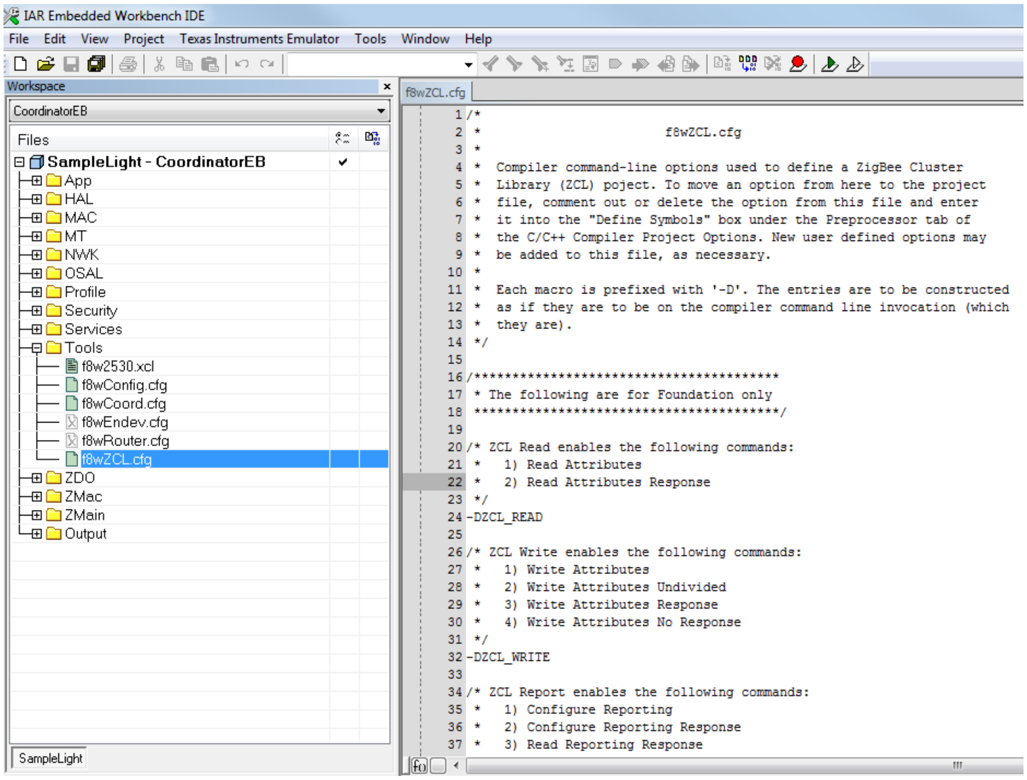
- Make sure all development software tools have been installed
- Consult section 2.2 to Power up the board.
- If Windows prompts to install a device driver, don’t let it connect to Windows Update. Instead, let Windows try to find the required driver automatically. If that fails, browse to: *C:\Program Files\IAR Systems\<Embedded Workbench>\<arm\8051>\drivers\Texas Instruments* to locate the necessary files.
- Select IAR project workspace (*.eww), and open it with IAR. Select ZigBee device type configuration from the *Workspace* pull-down menu. Refer to section 2.4 to help identify device type for each sample application. For instance, SampleLight is associated with a ZC (select **CoordinatorEB** from the Workspace pull-down menu).



```
/*****************************************************************************
  Filename:       zcl_sampleLight.c
  Revised:        $Date: 2012-12-11 15:32:29 -0800 (Tue, 11 Dec 2012) $
  Revision:       $Revision: 32568 $


  Description:    Zigbee Cluster Library - sample device application.


  Copyright 2006-2012 Texas Instruments Incorporated. All rights reserved.

  IMPORTANT: Your use of this Software is limited to those specific rights
  granted under the terms of a software license agreement between the user
  who downloaded the software, his/her employer (which must be your employer)
  and Texas Instruments Incorporated (the "License").  You may not use this
  Software unless you agree to abide by the terms of the License. The License
  limits your use, and you acknowledge, that the Software may not be modified,
  copied or distributed unless embedded on a Texas Instruments microcontroller
  or used solely and exclusively in conjunction with a Texas Instruments radio
  frequency transceiver, which is integrated into your product.  Other than for
  the foregoing purpose, you may not use, reproduce, copy, prepare derivative
  works of, modify, distribute, perform, display or sell this Software and/or
  its documentation for any purpose.

  YOU FURTHER ACKNOWLEDGE AND AGREE THAT THE SOFTWARE AND DOCUMENTATION ARE
  PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED,
  INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY, TITLE,
  NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL
  TEXAS INSTRUMENTS OR ITS LICENSORS BE LIABLE OR OBLIGATED UNDER CONTRACT,
  NEGLIGENCE, STRICT LIABILITY, CONTRIBUTION, BREACH OF WARRANTY, OR OTHER
  LEGAL EQUITABLE THEORY ANY DIRECT OR INDIRECT DAMAGES OR EXPENSES
  INCLUDING BUT NOT LIMITED TO ANY INCIDENTAL, SPECIAL, INDIRECT, PUNITIVE
  OR CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF PROCUREMENT
  OF SUBSTITUTE GOODS, TECHNOLOGY, SERVICES, OR ANY CLAIMS BY THIRD PARTIES
  (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), OR OTHER SIMILAR COSTS.

  Should you have any questions regarding your right to use this Software,
```

Figure 16: Selecting Correct Configuration

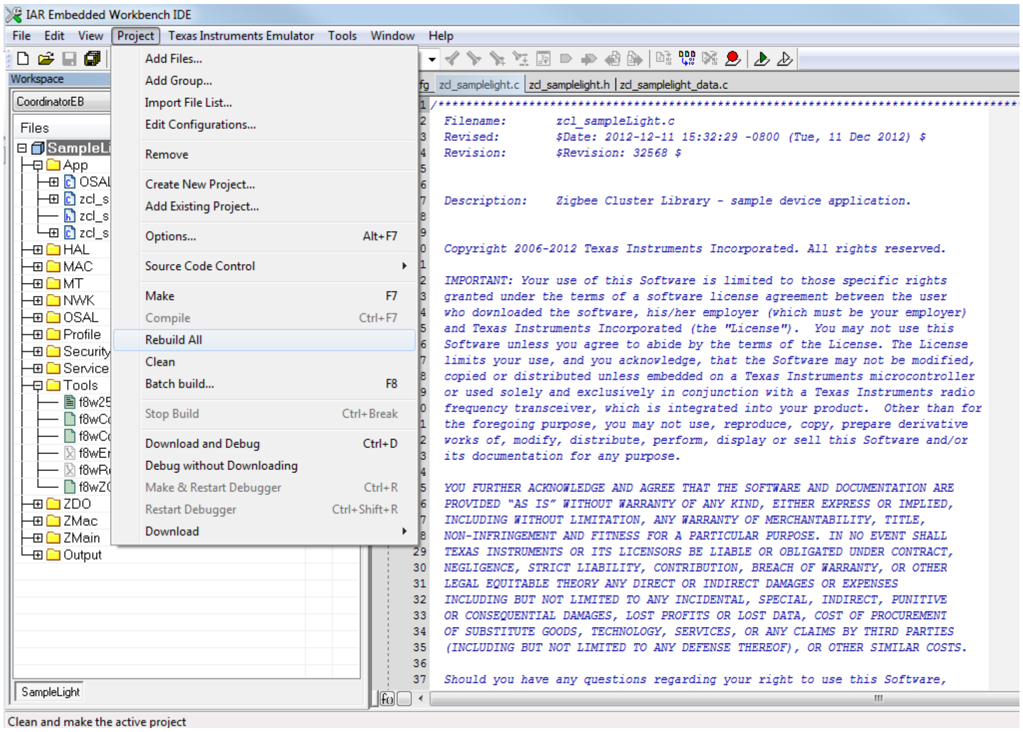
- Each sample application requires different set of clusters. By default, required application clusters are enabled in IAR project options. To enable specific clusters, consult f8wZCL.cfg file and enable it in IAR compiler options (**Figure 14**)



```
/*
 *                      f8wZCL.cfg
 *
 *  Compiler command-line options used to define a ZigBee Cluster
 *  Library (ZCL) poject. To move an option from here to the project
 *  file, comment out or delete the option from this file and enter
 *  it into the "Define Symbols" box under the Preprocessor tab of
 *  the C/C++ Compiler Project Options. New user defined options may
 *  be added to this file, as necessary.
 *
 *  Each macro is prefixed with '-D'. The entries are to be constructed
 *  as if they are to be on the compiler command line invocation (which
 *  they are).
 */

/****************************************
 * The following are for Foundation only
 ****************************************/

/* ZCL Read enables the following commands:
 *   1) Read Attributes
 *   2) Read Attributes Response
 */
-DZCL_READ

/* ZCL Write enables the following commands:
 *   1) Write Attributes
 *   2) Write Attributes Undivided
 *   3) Write Attributes Response
 *   4) Write Attributes No Response
 */
-DZCL_WRITE

/* ZCL Report enables the following commands:
 *   1) Configure Reporting
 *   2) Configure Reporting Response
 *   3) Read Reporting Response
```

Figure 17: ZCL Cluster configuration file for reference

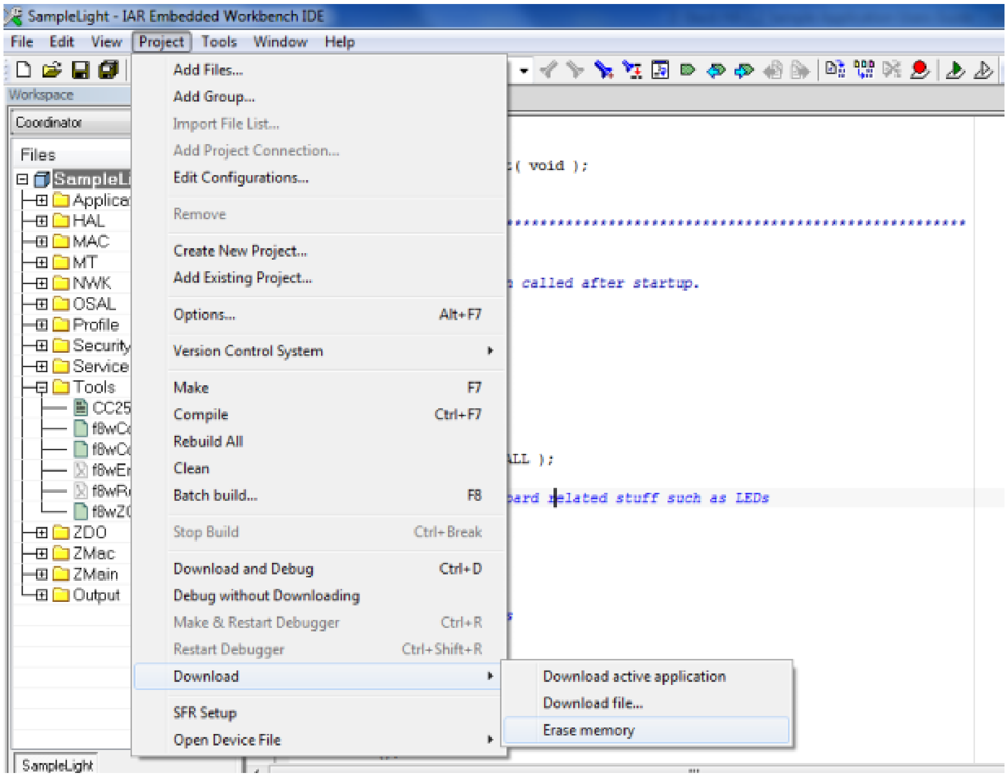
- Build the application by pulling down the *Project* menu and clicking on **Rebuild All**:



```
/*****************************************************************************
  Filename:       zcl_sampleLight.c
  Revised:        $Date: 2012-12-11 15:32:29 -0800 (Tue, 11 Dec 2012) $
  Revision:       $Revision: 32568 $


  Description:    Zigbee Cluster Library - sample device application.


  Copyright 2006-2012 Texas Instruments Incorporated. All rights reserved.

  IMPORTANT: Your use of this Software is limited to those specific rights
  granted under the terms of a software license agreement between the user
  who downloaded the software, his/her employer (which must be your employer)
  and Texas Instruments Incorporated (the "License").  You may not use this
  Software unless you agree to abide by the terms of the License. The License
  limits your use, and you acknowledge, that the Software may not be modified,
  copied or distributed unless embedded on a Texas Instruments microcontroller
  or used solely and exclusively in conjunction with a Texas Instruments radio
  frequency transceiver, which is integrated into your product.  Other than for
  the foregoing purpose, you may not use, reproduce, copy, prepare derivative
  works of, modify, distribute, perform, display or sell this Software and/or
  its documentation for any purpose.

  YOU FURTHER ACKNOWLEDGE AND AGREE THAT THE SOFTWARE AND DOCUMENTATION ARE
  PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED,
  INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY, TITLE,
  NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL
  TEXAS INSTRUMENTS OR ITS LICENSORS BE LIABLE OR OBLIGATED UNDER CONTRACT,
  NEGLIGENCE, STRICT LIABILITY, CONTRIBUTION, BREACH OF WARRANTY, OR OTHER
  LEGAL EQUITABLE THEORY ANY DIRECT OR INDIRECT DAMAGES OR EXPENSES
  INCLUDING BUT NOT LIMITED TO ANY INCIDENTAL, SPECIAL, INDIRECT, PUNITIVE
  OR CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF PROCUREMENT
  OF SUBSTITUTE GOODS, TECHNOLOGY, SERVICES, OR ANY CLAIMS BY THIRD PARTIES
  (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), OR OTHER SIMILAR COSTS.

  Should you have any questions regarding your right to use this Software,
```

Figure 18: Building Sample Application

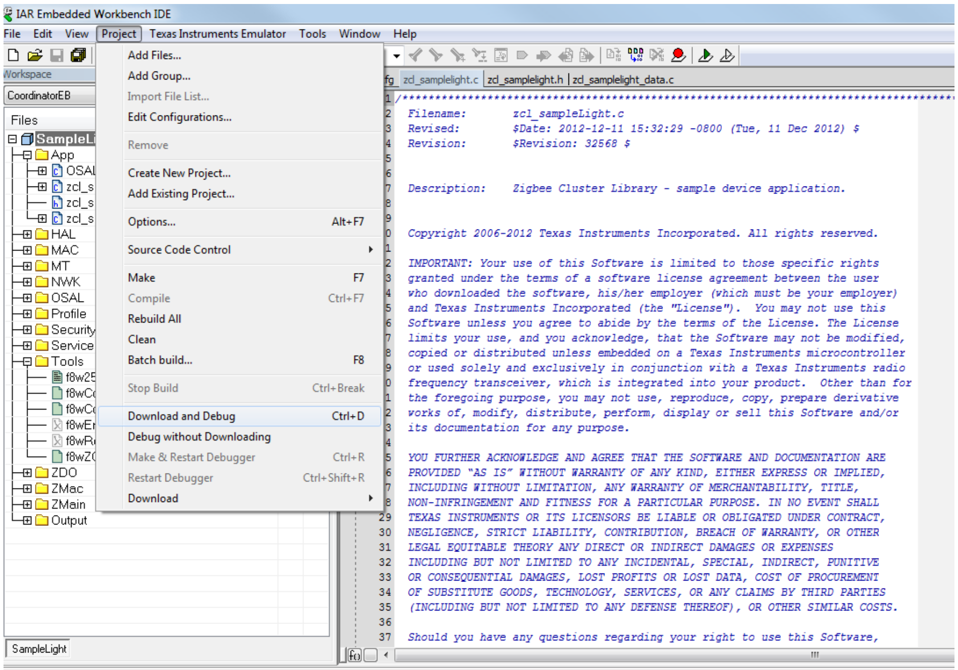
- In the case of CC2538, erase flash prior to programming new image:



```
( void );
*************************************************************
 called after startup.
LL );
ard related stuff such as LEDs
```

Figure 19: Erase Memory in CC2538

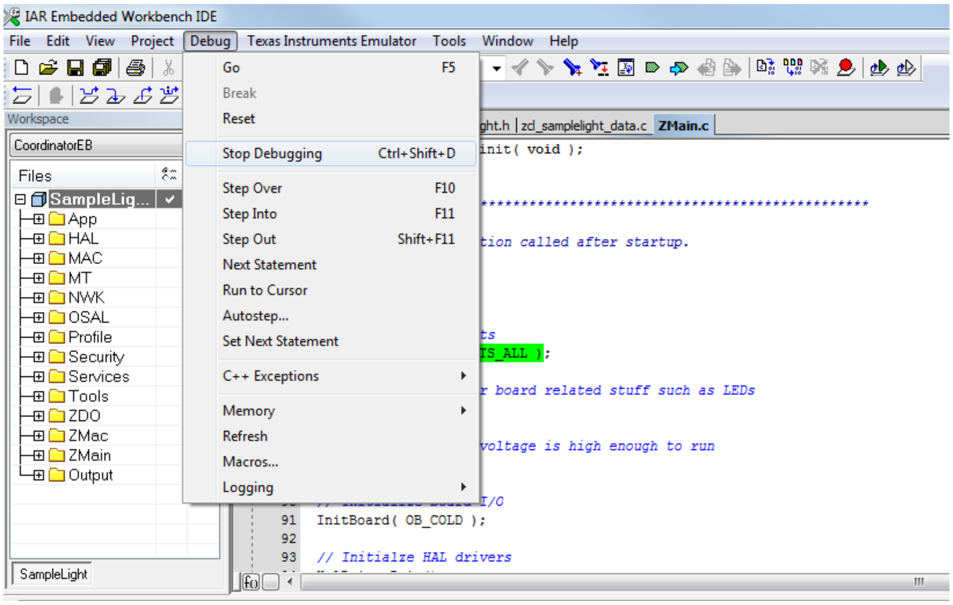
- Program (download) the sample application into the module by pulling down the *Project* menu and clicking on **Download and Debug**:



```
/*****************************************************************************
  Filename:       zcl_sampleLight.c
  Revised:        $Date: 2012-12-11 15:32:29 -0800 (Tue, 11 Dec 2012) $
  Revision:       $Revision: 32568 $


  Description:    Zigbee Cluster Library - sample device application.


  Copyright 2006-2012 Texas Instruments Incorporated. All rights reserved.

  IMPORTANT: Your use of this Software is limited to those specific rights
  granted under the terms of a software license agreement between the user
  who downloaded the software, his/her employer (which must be your employer)
  and Texas Instruments Incorporated (the "License").  You may not use this
  Software unless you agree to abide by the terms of the License. The License
  limits your use, and you acknowledge, that the Software may not be modified,
  copied or distributed unless embedded on a Texas Instruments microcontroller
  or used solely and exclusively in conjunction with a Texas Instruments radio
  frequency transceiver, which is integrated into your product.  Other than for
  the foregoing purpose, you may not use, reproduce, copy, prepare derivative
  works of, modify, distribute, perform, display or sell this Software and/or
  its documentation for any purpose.

  YOU FURTHER ACKNOWLEDGE AND AGREE THAT THE SOFTWARE AND DOCUMENTATION ARE
  PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED,
  INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY, TITLE,
  NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL
  TEXAS INSTRUMENTS OR ITS LICENSORS BE LIABLE OR OBLIGATED UNDER CONTRACT,
  NEGLIGENCE, STRICT LIABILITY, CONTRIBUTION, BREACH OF WARRANTY, OR OTHER
  LEGAL EQUITABLE THEORY ANY DIRECT OR INDIRECT DAMAGES OR EXPENSES
  INCLUDING BUT NOT LIMITED TO ANY INCIDENTAL, SPECIAL, INDIRECT, PUNITIVE
  OR CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF PROCUREMENT
  OF SUBSTITUTE GOODS, TECHNOLOGY, SERVICES, OR ANY CLAIMS BY THIRD PARTIES
  (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), OR OTHER SIMILAR COSTS.

  Should you have any questions regarding your right to use this Software,
```

Figure 20: 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**:



```
InitBoard( OB_COLD );

// Initialze HAL drivers
```

Figure 21: Stop Debugging

- Board must be reset after the code image is downloaded. Switch it off and on.

- Repeat this process for the SampleSwitch and other sample applications. Refer to **section 2.4** to verify associated applications.

3.1.2 CC2630/CC2650 and SensorTag

The CC2630/CC2650 workspace is different from the CC2538 workspace, it consists of 2 parts:

1. **ZStackCore-<Device Type>**: IAR Project for building the Z-Stack Core Stack for a specific device type such as end device.
2. **<Sample application>-SmartRF06**: IAR Project for the building the Z-Stack Home Sample Application and RTOS.

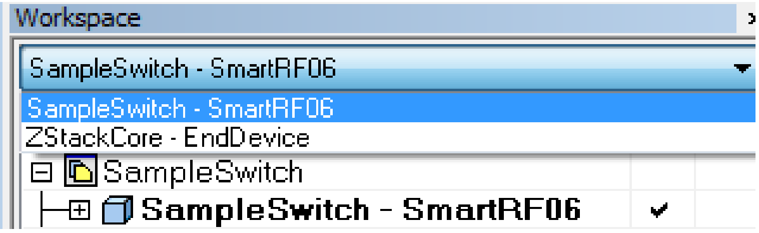


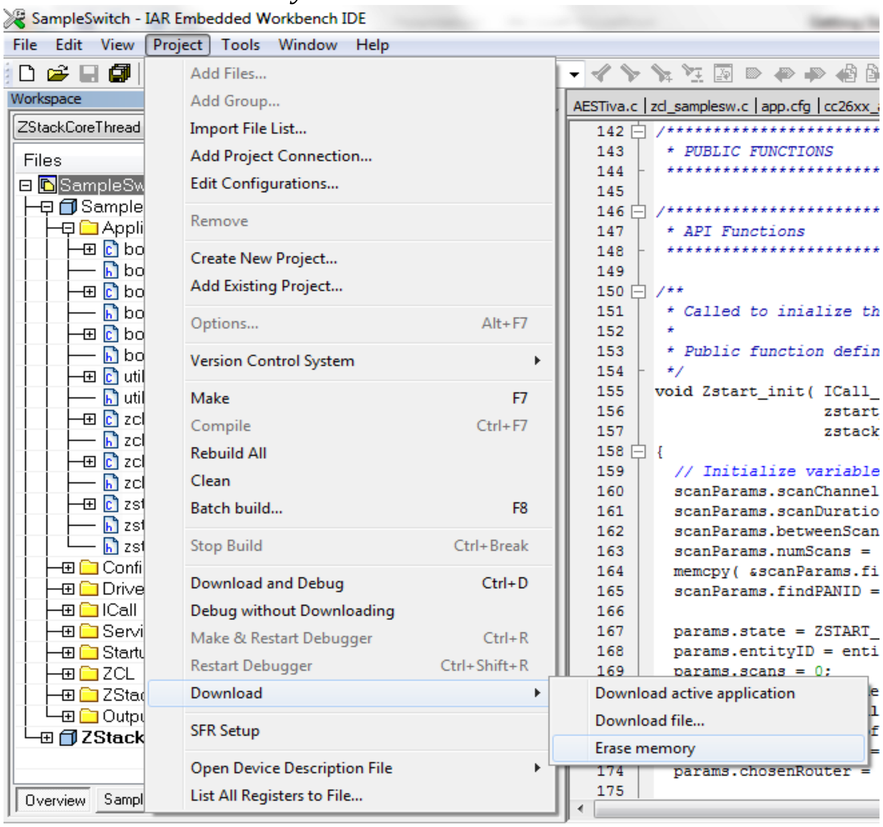
Figure 22 CC2630/CC2650 project configurations

This step-by-step process will use the SampleSwitch application as an example. The CC2630/CC2650 SampleSwitch application can be found in *C:\<Home Automation Install dir>\ Projects\zstack\HomeAutomation\SampleSwitch\CC26xx*

1 Connect the SmartRF06 with the CC2630/CC2650EM to the PC and power on the SmartRF06EB.

2 To use the Sensortag instead of the CC2630/CC2650EM, connect the SensorTag to the SmartRF06 board using a ribbon cable as shown in Figure 13.

3 Erase the memory of the CC2630/CC2650 by going to the menu *Project→Download→Erase Memory*.



```
/************************
 * PUBLIC FUNCTIONS
 ************************

/************************
 * API Functions
 ************************

/**
 * Called to inialize th
 *
 * Public function defin
 */
void Zstart_init( ICall_
                  zstart
                  zstack
{
  // Initialize variable
  scanParams.scanChannel
  scanParams.scanDuratio
  scanParams.betweenScan
  scanParams.numScans =
  memcpy( &scanParams.fi
  scanParams.findPANID =

  params.state = ZSTART_
  params.entityID = enti
  params.scans = 0;
  params.chosenRouter =
```

Figure 23 Erase memory

4 After erasing the memory, select *ZStackCore-EndDevice* as the active project using the drop down menu.

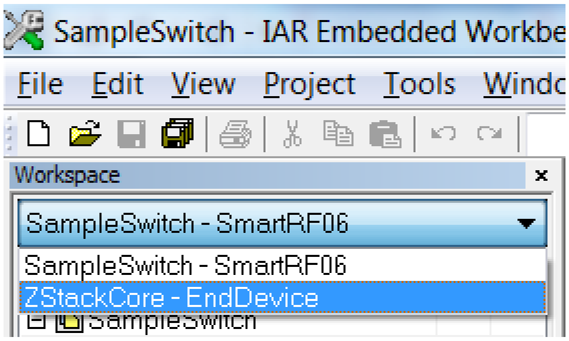
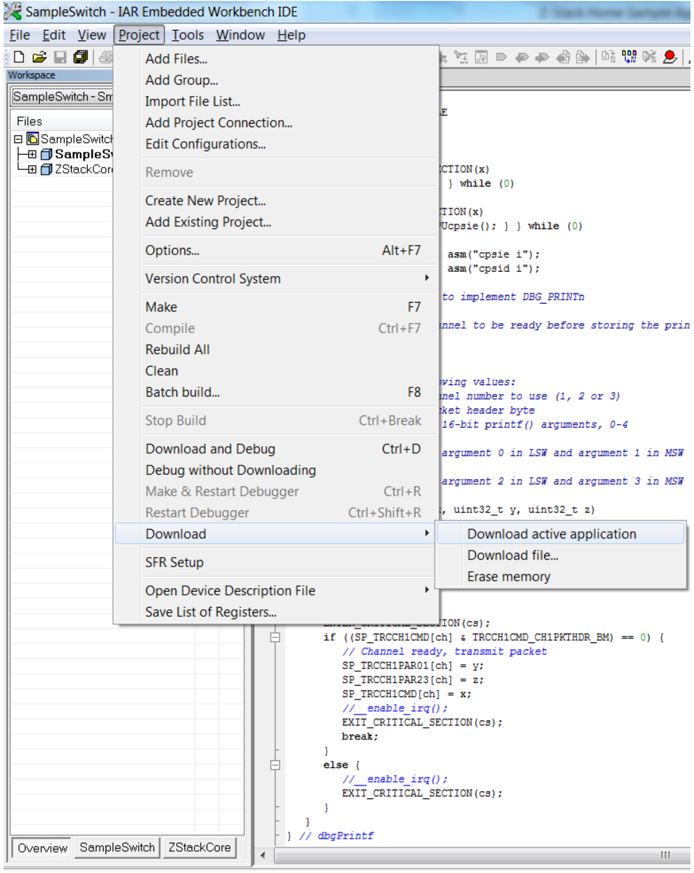


Figure 24 Workspace drop down menu

5 Download the *ZStackCore* application to the CC2630/CC2650 by going to the menu *Project →Download →Download Active Application*.



```
      if ((SP_TRCCH1CMD[ch] & TRCCH1CMD_CH1PKTHDR_BM) == 0) {
        // Channel ready, transmit packet
        SP_TRCCH1PAR01[ch] = y;
        SP_TRCCH1PAR23[ch] = z;
        SP_TRCCH1CMD[ch] = x;
        //__enable_irq();
        EXIT_CRITICAL_SECTION(cs);
        break;
      }
      else {
        //__enable_irq();
        EXIT_CRITICAL_SECTION(cs);
      }
    }
} // dbgPrintf
```

Figure 25 Download active application menu

6 Switch to *SampleSwitch – SmartRF06* project workspace by using the drop down menu.

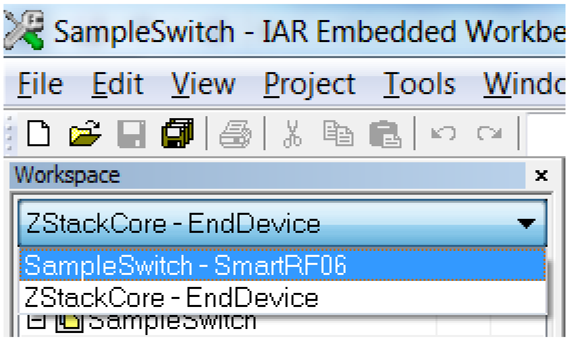
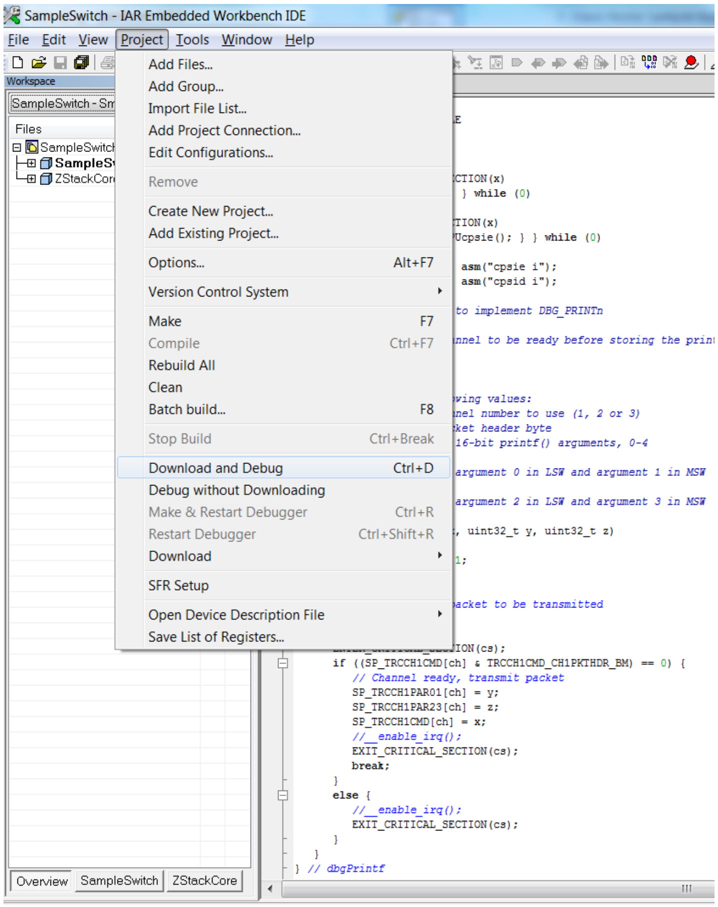


Figure 26 Workspace drop down menu

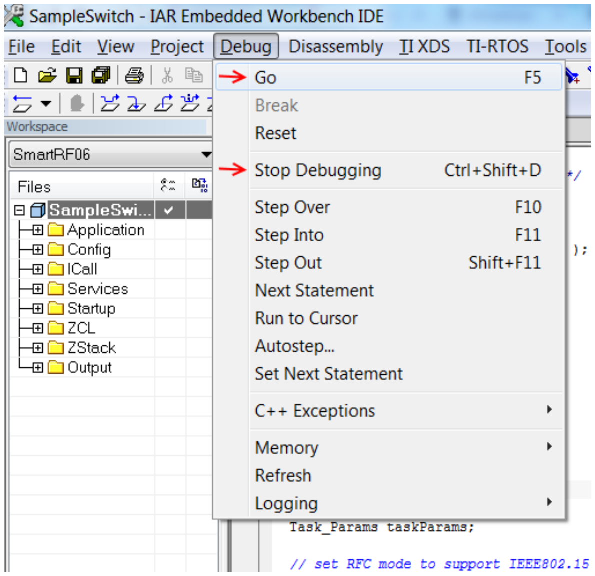
7 Download the *SampleSwitch – SmartRF06* application to the CC2630/CC2650 by going to the menu *Project→Download and Debug*.



```
      if ((SP_TRCCH1CMD[ch] & TRCCH1CMD_CH1PKTHDR_BM) == 0) {
        // Channel ready, transmit packet
        SP_TRCCH1PAR01[ch] = y;
        SP_TRCCH1PAR23[ch] = z;
        SP_TRCCH1CMD[ch] = x;
        //__enable_irq();
        EXIT_CRITICAL_SECTION(cs);
        break;
      }
      else {
        //__enable_irq();
        EXIT_CRITICAL_SECTION(cs);
      }
    }
} // dbgPrintf
```

Figure 27 Download and debug

8 Now that the CC2630/CC2650 has been flashed, go to the menu *Debugging* → *Go* to start debugging or *Debugging* → *Stop Debugging* to stop debugging.



3.2 Running Sample Applications

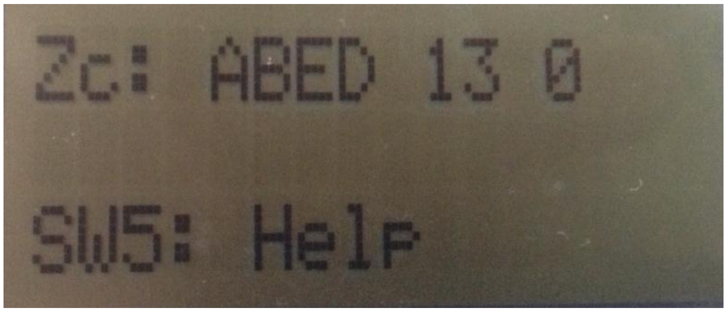
After compiling and programming the boards with their appropriate sample applications, one can use these applications as follows:

3.2.1 SampleLight, SampleSwitch

Traditionally, in ZigBee Home Automation networks, the lights are configured as Coordinator/ Routers, while the switches are configured as End Devices. This is due to the fact that the lights would anyhow require a constant power source (for the actual light bulb), while the switches can be battery operated.

Note, the following sections do not apply to the Zlight2 platform as it does not have an LCD. Also note that the ZLight2 Sample Light will self-initiate EZMode upon power up.

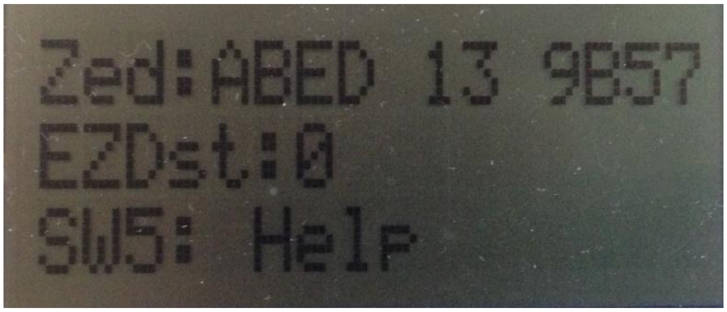
1. Power on SampleLight and SampleSwitch boards. Once powered up, LCD will show “Texas Instruments”, the MAC address, and the device name. When the coordinator has formed a network, screen will appear as it does in Figure below. Devices will be ready for EZ-Mode pairing.



```
Zc: ABED 13 0

SW5: Help
```

Figure 28: ZigBee Coordinator Startup screen

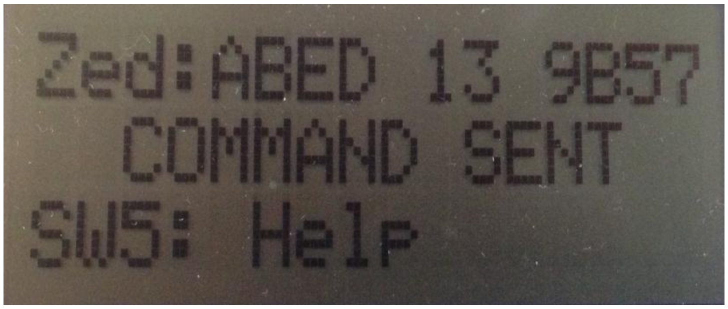
2. To pair the devices using EZ-Mode, press SW2 on both devices; after which, EZMODE will display on the 2nd line of the LCD screen. If pairing is successful, the 2nd line will display: “EZMODE: SUCCESS”. If the device is the EZ-Mode “initiator”, the “target” device’s network information will be displayed on the initiator’s 2nd line. LED1 will blink throughout this process until EZ-Mode pairing has finished. EZMODE device’s information or status will disappear after few seconds.



```
Zed:ABED 13 9B57
EZDst:0
SW5: Help
```

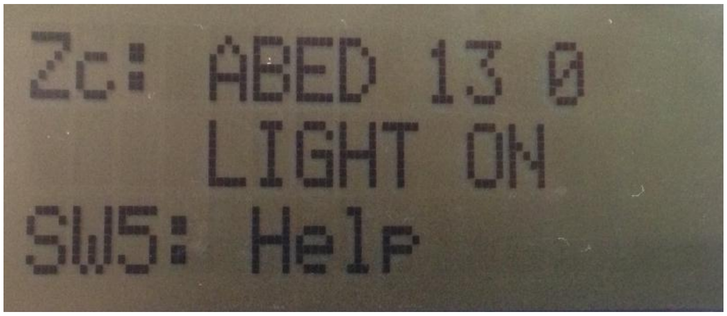
Figure 29: Initiator pairing with the Target

3. Once the devices have been successfully paired, on/off light toggle command can be sent from the SampleSwitch to the SampleLight using SW1.



```
Zed:ABED 13 9B57
  COMMAND SENT
SW5: Help
```

Figure 30: SampleSwitch sending Toggle command



```
Zc: ABED 13 0
    LIGHT ON
SW5: Help
```

Figure 31: SampleLight updating Light status

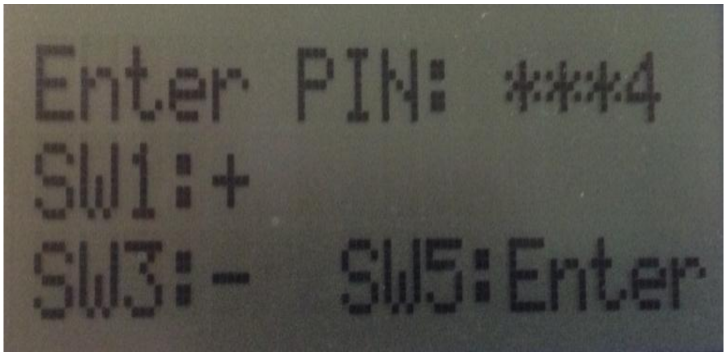
3.2.2 Sample SampleDoorLock, SampleDoorLockController

1. Power on the SampleDoorLock and SampleDoorLockController boards. Once powered up, LCD will show “Texas Instruments”, the MAC address, and the device name. When the coordinator has formed a network, screen will appear as shown in **Figure 28** and the devices will be ready for EZ-Mode pairing.

2. To pair the devices using EZ-Mode, press SW2 on both devices; after which, EZMODE will display on the 2nd line of the LCD screen. If pairing is successful, the 2nd line will display: “EZMODE: SUCCESS”. If the device is the EZ-Mode “initiator”, the “target” device’s information will be displayed on the initiator’s 2nd line. LED1 will blink throughout this process until EZ-Mode pairing has finished. Refer to **Figure 29**. EZMODE device’s information or status will disappear after few seconds.

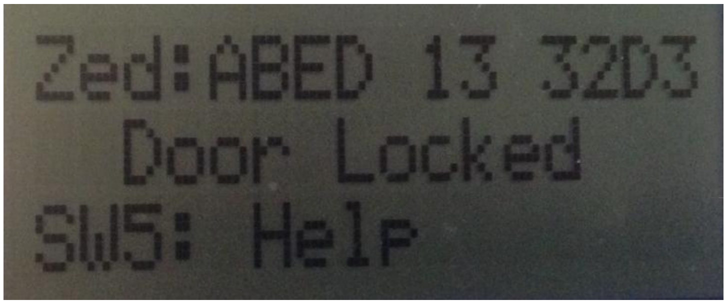
3. Once the devices have been successfully paired, door lock toggle command can be sent from the controller to the door lock by pressing SW1 (**Figure 33**)

4. Master PIN is stored on the SampleDoorLock. Both devices have a default PIN of “1234”. This PIN must match in order for the SampleDoorLockController to successfully lock/unlock the SampleDoorLock. To change the default pin, consult section 2.4.2. **Figure 32** shows change pin snapshot. If the PIN sent to the SampleDoorLock is invalid, the display will appear as shown in **Figure 34**



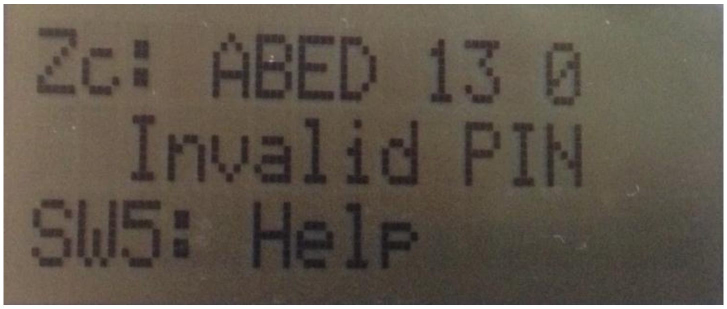
Enter PIN: ***4
SW1:+
SW3:- SW5:Enter

Figure 32: Setting SampleDoorLock PIN



Zed:ABED 13 32D3
 Door Locked
SW5: Help

Figure 33: SampleDoorLockController showing status of SampleDoorLock



Zc: ABED 13 0
 Invalid PIN
SW5: Help

Figure 34: SampleDoorLock showing Invalid PIN update

3.2.3 SampleThermostat, SampleTemperatureSensor, SampleHeatingCoolingUnit

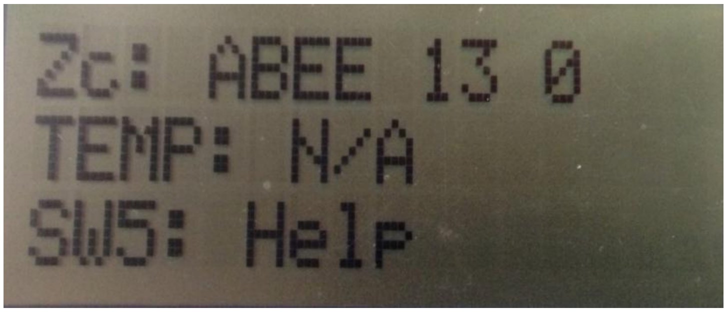
1. Power on the SampleThermostat, SampleTemperatureSensor, and SampleHeatingCoolingUnit boards. Once powered up, LCD will show “Texas Instruments”, the MAC address, and the device name. When the coordinator has formed a network, the screen will appear as shown in **Figure 28** and the devices will be ready for EZ-Mode pairing.

2. To pair the devices using EZ-Mode, press SW2 on the SampleThermostat and SampleHeatingCoolingUnit; after which, EZMODE will display on the 2nd line of the LCD screen. Repeat the pairing process for the SampleThermostat and SampleTemperatureSensor. If pairing is successful, 2nd line will display: “EZMODE: SUCCESS”. If the device is the EZ-Mode “initiator”, the “target” device’s information will be displayed on the initiator’s 2nd line. LED1 will blink throughout this process until

EZ-Mode pairing has finished. Refer to **Figure 29.** EZMODE device's information or status will disappear after few seconds.

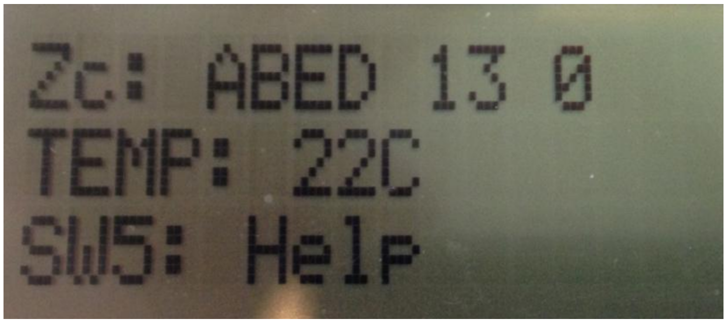
3. Once all three devices have been successfully paired, current temperature reading will be sent from the SampleTemperatureSensor to the SampleThermostat every 10 seconds.

4. Refer to section 2.4.3 for commands. On the SampleTemperatureSensor, increase or decrease the temperature reading using SW1/SW3. LED2 will turn on for temperatures above 20° C. LED1 will turn on for temperatures below 24° C. This will update on the SampleThermostat every 10 seconds, and also when the user changes the temperature on sensor



```
Zc: ABEE 13 0
TEMP: N/A
SW5: Help
```

Figure 35: SampleThermostat prior to receiving temperature readings

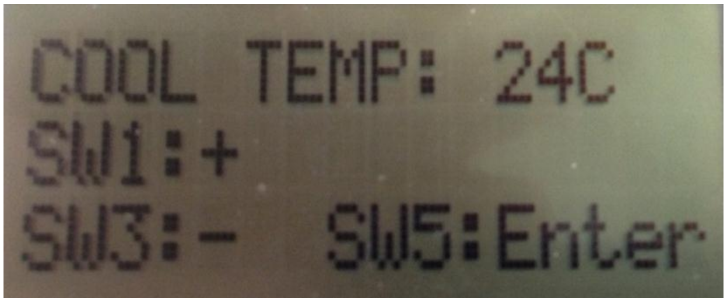


```
Zc: ABED 13 0
TEMP: 22C
SW5: Help
```

Figure 36: SampleThermostat updating temperature as reported by SampleTemperatureSensor

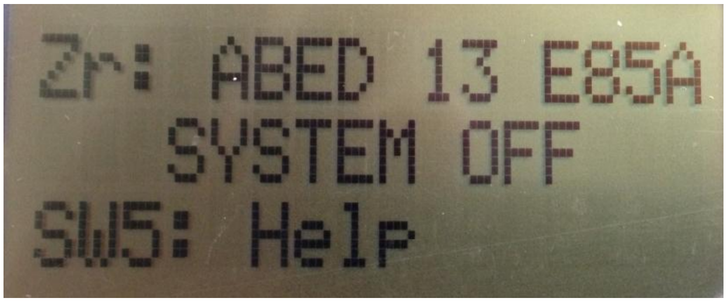
5. Default heating/cooling setpoints on Thermostat can be changed by the user. Consult section 2.4.3 for commands. Heating or Cooling modes are indicated by Thermostat to SampleHeatingCoolingUnit when to turn on the heating or cooling cycle. LEDs on the SampleThermostat will reflect what state the SampleHeatingCoolingUnit is in, based on the heating and cooling setpoints. LED1 indicates a cooling cycle, LED2 indicates a heating cycle. These LED attributes reflect across the SampleHeatingCoolingUnit as well.

6. SampleHeatingCoolingUnit relies on commands sent from the SampleThermostat that include: heating cycle, cooling cycle, or system off. By default, this unit is in the off state.



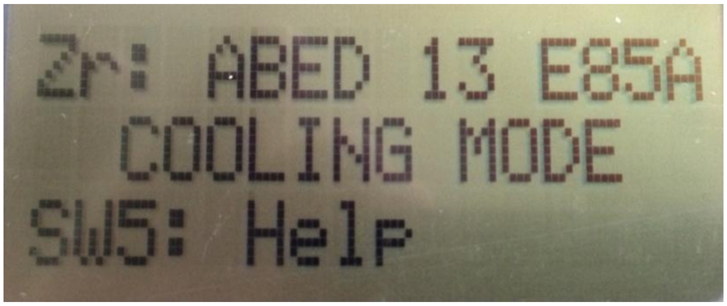
```
COOL TEMP: 24C
SW1:+
SW3:-  SW5:Enter
```

Figure 37: Setting Cooling Setpoint on SampleThermostat



```
Zr: ABED 13 E85A
   SYSTEM OFF
SW5: Help
```

Figure 38: SampleHeatingCoolingUnit in OFF Mode



```
Zr: ABED 13 E85A
  COOLING MODE
SW5: Help
```

Figure 39: SampleHeatingCoolingUnit in Cooling Mode

3.2.4 SensorTag

The SensorTag sample application has two different usage modes; it can be used as a switch or a temperature sensor. These three modes are integrated into a single sample application, the steps below explain how to use the SampleSensorTag as a switch in combination with a SampleLight, the same approach can be used for the temperature sensor.

1. Power on the SampleLight in one of the supported platforms and the SensorTag by inserting a coin cell battery or by connecting it to the SmartRF06board as shown in Figure 23. Once powered up, the LCD, in the SampleLight board, will show “Texas Instruments”, the MAC address, and the device name. When the coordinator has formed a network, in this case the SampleLight, the screen will appear as shown in Figure 28 and the devices will be ready for EZ-Mode pairing.

2. To pair the devices using EZ-Mode, press SW2 on the SampleLight and SW1 on the SensorTag; after which, EZMODE will display on the 2nd line of the SmartRF06 LCD screen. If pairing is successful, the 2nd line will display: “EZMODE: SUCCESS” and the LED in the SensorTag will light up.

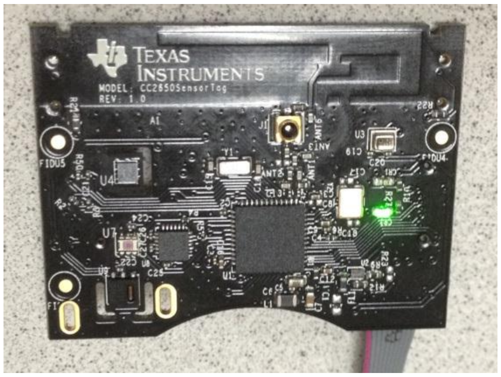


Figure 40 Successful network join

3. Once all devices have been successfully paired, on/off light toggle command can be sent from the SensorTag to the SampleLight by pressing SW2.

3.3 Preserving Network Configuration

NV_RESTORE is not turned on by default in CC2530 and CC2538 sample applications, so if any of the devices is power cycled, it will need to redo network joining. In production devices, NV_INIT and NV_RESTORE shall be specified, so the configuration will be saved in NV and the user will not need to repeat the initial setup after each power cycle. It is also recommended for production devices that should not require user intervention to join the network, to disable HOLD_AUTO_START, so the device starts automatically.

When NV is enabled, you will notice that following the Switch joining the network, the Light’s LCD will show “Orphan Response Sent”. This is normal behavior – When the Switch is connecting to a previously configured network, it starts with sending Orphan Notification”.

3.4 Selecting the Channel of Operation on CC2630/CC2650

To change the channel of operation or to specify a channel list, open the file znwk_config.h within the application directory in the IAR project workspace.

Set the channel by defining the ZSTART_DEFAULT_CHANLIST appropriately as shown in the example below which is setting the channel of operation to 24.

```
#define ZSTART_DEFAULT_CHANLIST 0x01000000  // 24 - 0x18
```

4 HA Secure Joining

The HA Profile requires that all devices share a default Trust Center Link Key and that the network key is delivered to joining devices secured with that link key.

The Default Trust Center Link Key is specified in the ZigBee Home Automation Public Application Profile as:

0x5A 0x69 0x67 0x42 0x65 0x65 0x41 0x6C 0x6C 0x69 0x61 0x6E 0x63 0x65 0x30 0x39
(listed in little-endian format).

You will need to configure the Sniffer software with the Default Trust Center Link Key. To configure Ubiqua follow these steps within Ubiqua GUI:

1. Select Tools → Options
2. Under Security in the Keychain Tab click Add
3. Select Type: Application or Trust Center Link Key
4. Enter the provided key in the 'key' field, using the following format:
   5A:69:67:42:65:65:41:6C:6C:69:61:6E:63:65:30:39
5. Add and then OK to confirm

Ubiqua has the ability to extract the Network Key from the APS Transport Key message (which is used to transfer the Network key to the joining device). Nevertheless, in some situations, the Network Key should be manually configured in order for Ubiqua to decrypt packets: One example is network-encrypted packets which are sent by the coordinator before a Transport Key message is sent (i.e. before any device joins the network), such as Link Status messages. Another example is when sniffing is started after the device has already joined the network, so no Transport Key messages will be captured. Manual configuration of the Network Key is done in a way similar to the Trust Center Link Key (see above), only that the key type is now 'ZigBee Network Key', and the key itself is 01:03:05:07:09:0B:0D:0F:00:02:04:06:08:0A:0C:0D. This is the default network key. It can be changed in f8wConfig_HA.cfg / DEFAULT_KEY.
There are basically two joining scenarios for a HA Profile device:

4.1 Joining directly to the Trust Center

When a device joins the network, and its parent is the Trust Center, the transport key command is encrypted with the default Trust Center Link key, and sent to it directly by the Trust Center:

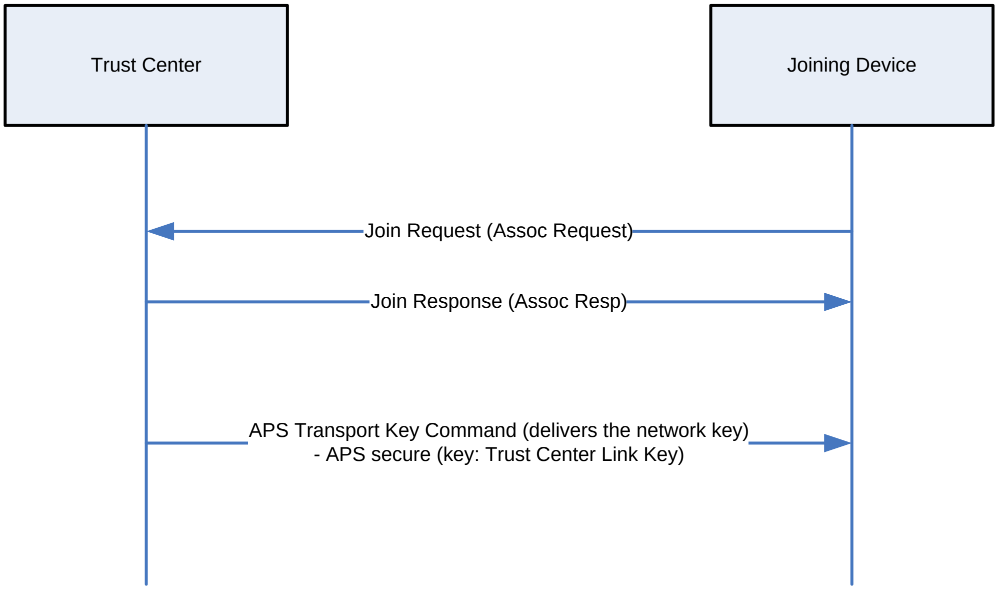

Trust Center
Joining Device
Join Request (Assoc Request)
Join Response (Assoc Resp)
APS Transport Key Command (delivers the network key)
- APS secure (key: Trust Center Link Key)

Figure 41: HA Joining directly to the Trust Center

4.2 Joining via an intermediate router

When a device joins the network, but its parent isn’t the Trust Center, the transport key command is tunneled from the Trust Center, through the parent of the joining device, to the joining device:

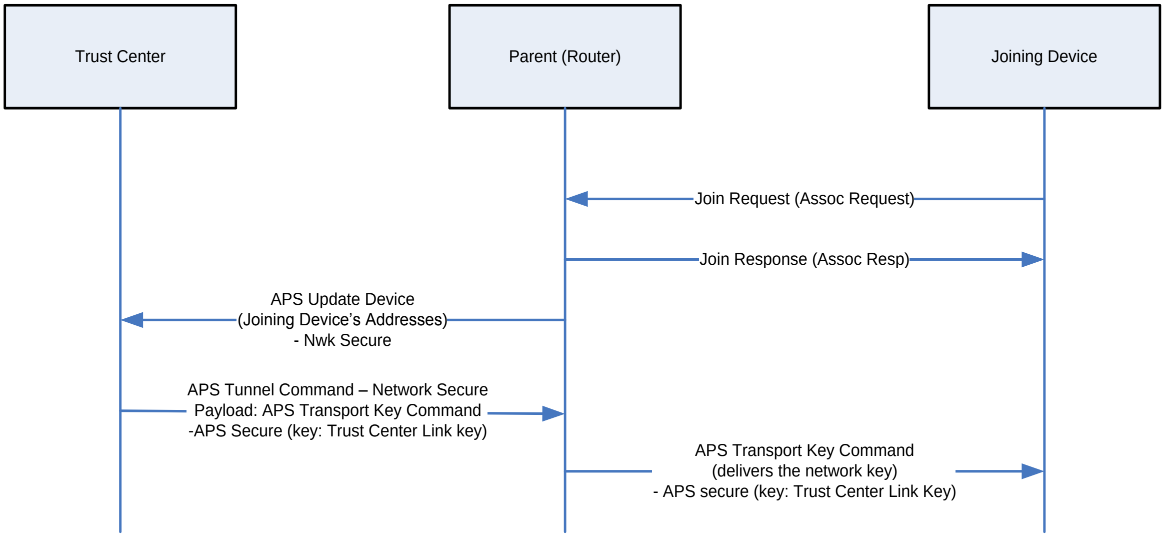

Trust Center
Parent (Router)
Joining Device
Join Request (Assoc Request)
Join Response (Assoc Resp)
APS Update Device
(Joining Device’s Addresses)
- Nwk Secure
APS Tunnel Command – Network Secure
Payload: APS Transport Key Command
-APS Secure (key: Trust Center Link key)
APS Transport Key Command
(delivers the network key)
- APS secure (key: Trust Center Link Key)

Figure 42: Joining the HA network via intermediate router

When viewing the traffic with a ZigBee packet sniffer, you will notice that in practice, the Update Device command is sent twice: once without APS encryption (Only NWK encryption is used), and once with APS encryption applied. This behavior is for backward compatibility with devices that does not support reception of APS encrypted Update Device command. For each of the two Update Device commands, the resulting Tunnel command is APS encrypted.

4.3 Disabling security

Secure Joining is enabled by default, as mandated by the ZigBee Home Automation specification. To disable it, e.g. for debug purpose, set SECURE=0 in f8wConfig.cfg and remove the definition of TC_LINKKEY_JOIN from Project→Options→C/C++ Compiler → Preprocessor → Defined symbols.

Security shall be either enabled or disabled by all the devices in the network (Secured and Unsecured devices cannot be mixed in the same network).