Power Consumption Measurements and Optimization for CC2538 End Device With Z-Stack

Suyash Jain

ABSTRACT

This application report describes how to set up power consumption measurements for a CC2538EM (end-device) node running Z-Stack™. The application used for these measurements is the Z-Stack Home sample light and sample switch application, included in the Z-Stack-Home release (downloaded from [3] and it is described in [4]). This document uses the ZigBee PRO stack profile.

The spreadsheet discussed in this application report can be downloaded from the following URL: http://www.ti.com/lit/zip/swra456.

Contents

List of Figures

List of Tables

1 Introduction

The CC2538 is the ideal SoC for high-performance ZigBee applications. It combines a powerful ARM Cortex M3-based MCU system with up to 32K on-chip RAM and up to 512 K on-chip flash with a robust IEEE 802.15.4 radio. This enables it to handle complex network stacks with security, demanding applications such as networks requiring large number of zigbee nodes, and over-the-air download.

Thirty-two GPIOs and serial peripherals enable simple connections to the rest of the board. The powerful security accelerators enable quick and efficient authentication and encryption while leaving the CPU free to handle application tasks. The low-power modes with retention enable quick startup from sleep and minimum energy spent to perform periodic tasks. For a smooth development, the CC2538 includes a powerful debugging system and a comprehensive driver library. To reduce the application flash footprint, CC2538 ROM includes a utility function library and a serial boot loader. Combined with the free to use ZigBee stack from Texas Instruments, the CC2538 provides the most capable and robust ZigBee solution in the market

For similar measurements on CC2530, see the application note [15].

The measurement setup consists of a ZigBee End Device (ZED) and a ZigBee Coordinator (ZC). The ZED will periodically poll the ZC for data, and in between the different poll messages the ZED goes to sleep (using Power Mode 2) to save power.

This document first describes which hardware and software (see Section 3) was used for the measurement setup, which is described in Section 4. The obtained results are shown and discussed in Chapter 5.

Note that there are many factors that influence the overall power consumption and that the results presented in this document should only be regarded as indicative for what is possible to achieve in systems with similar hardware.

For more information about the usage of the CC2538, you are referred to the CC2538 User's Guide [7]. More details regarding the power management using Z-Stack on the CC2538 can be found in [8]. For the examples and API's for using the peripheral modules on the CC2538, the USB device controller, and the examples on the drivers used to interface with external peripherals, you can download the software package *CC2538 Foundation Firmware* [14].

2 Abbreviations and Acronyms

Table 1. Abbreviations and Acronyms

ACK	Acknowledgement
APS	Application Support Sub-Layer
BB	Battery Board
CSMA-CA	Carrier Sense Multiple Access, Collision Avoidance
DB	Development Board
DK	Development Kit
EB	Evaluation Board
EW	Embedded Workbench
I	Current (Ampere, A)
mA	Milliampere (10^{-3} A)
MAC	Medium Access Control
OSAL	Operating System Abstraction Layer
PM2	Power Mode 2
PHY	Physical Layer
R	Resistance (Ohm, Ω)
RAM	Random Access Memory
RX	Receive
TX	Transmit
µA	Microampere (10^{-6} A)
V	Voltage (Volt)
ZC	ZigBee Coordinator
ZED	ZigBee End Device
ZR	ZigBee Router

3 System Overview

This chapter describes the hardware and software used for the measurement setup described in Section 4.

3.1 Hardware - CC2538DK

Figure 1 shows the hardware used in this application report (one SmartRF06EB, one CC2531-USB-dongle, two CC2538EMs, and cabling) is taken from the CC2538 Development Kit (CC2538DK). SmartRF06BB was used for the measurements.

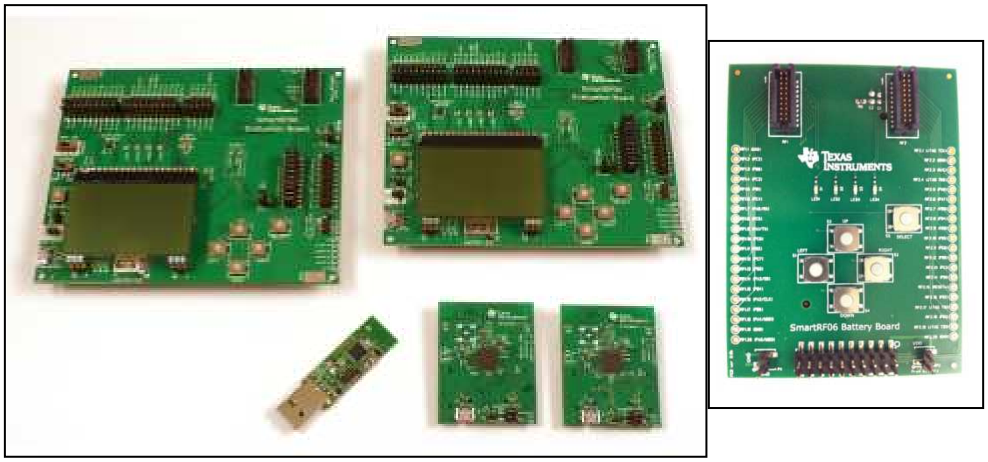

Figure 1. CC2538DK – HW Included in the CC2538 Development Kit and SmartRF06 Battery Board

Additionally, a DC power supply, a digital real time oscilloscope, a 2.8 Ω resistor, and some cabling are used. The detailed measurement setup is shown in Section 4.

3.2 Software – Z-Stack

The following subsections describe the software that was used to perform the measurements.

3.2.1 Z-Stack Development Environment

The measurement setup shown in Chapter 4 is based on the Z-Stack Home-1.2.0 Sample Switch and Sample Light applications, which comes as part of the Z-Stack-Home-1.2.0 release [3] and is described in [4].

In order to be able to compile the application and load onto the CC2538EMs, you must install the correct version of the IAR Embedded Workbench™ for Arm (mentioned in the readme file with the Z-Stack Home install) which can be obtained at http://www.iar.com/ti_zigbee (evaluation version) or from http://www.iar.com/downloads.

4 Measurement Setup

This section explains how to setup and configure the hardware and software described in Section 3 in order to produce the measurements shown and discussed in Chapter 5.

4.1 Instrumentation

The general idea of the current consumption measurement is to visualize the current profile on an oscilloscope by measuring the voltage drop over a fixed resistor. The set up is illustrated in Figure 2.

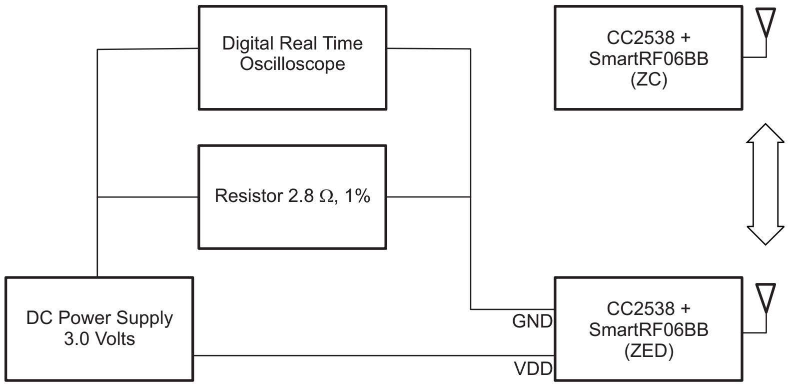

Figure 2. Measurement Setup

The oscilloscope provides a graphical representation of the voltage drop over the resistor. Since there is a linear relationship between voltage and current (Ohm's Law), the same graphical representation illustrates the current consumed by the system (see Section 4.2.4). The CC2538EM is connected as shown in Figure 2 via two Jumpers (GND and VDD on the SmartRF06 battery board). The power supply is setup to provide a constant voltage of 3.0 V.

NOTE: This measurement method may influence the result because of the voltage drop over the resistor and the stray (cable) resistance. Ideally, the measurement setup should have been verified with a high-precision ampere-meter. The results are also read from the oscilloscope using the eye, which also influences the accuracy. Keep in mind that the results shown in this application report are indicative and that you would have to perform your own measurements on your own hardware to know its real power consumption.

The very low-current consumption during sleep mode (PM2) has not been measured using the above setup as this setup does not allow measuring a small current in the magnitude of μA due to measurement accuracy. Instead, this current was measured to approximately 1.6 μA with an ampere-meter (replacing the resistor and oscilloscope with the ampere-meter in Figure 2), which is in accordance to the value stated in the CC2538 data sheet [1].

4.2 Software Setup

This section describes the software application setup for the measurements (obtained with the measurement setup shown in Figure 2) and how to reproduce it.

The applications programmed on the CC258EM modules are the Home Automation sample applications SampleSwitch and SampleLight using ZigBee-Pro. Both are included in the Z-Stack Home-1.2.0 release [3] and described in [4]. They are part of the sample application demonstrating the use of the ZigBee Home Automation profile. Further details about the usage of the Z-Stack and features like binding can be found in the ZigBee Developer's Guide [13].

The idea is to bind the light device (SampleLight – EndDeviceEB) to the switch (SampleSwitch – CoordinatorEB) as explained in Section 4.2.4. Typically, the binding operation is a one time operation in the lifetime of the device operation to find and store the information about the device it wants to talk to. Then the ZED polls its parent (the ZC) periodically to see whether there are pending messages for it. This polling operation is referred to as **Operation1** in the following when there is no data pending. The second scenario, **Operation2**, describes the situation where a message (“toggle the light”) is pending, received, processed, and acknowledged by the light application.

NOTE: In the described setup, the ZED (CC2538EM + SmartRF06BB) is running the SampleLight application and the ZC (CC2538EM + SmartRF06EB) is running the SampleSwitch application, even though it would seem more realistic with the switch as a battery-powered ZED instead of the light. The setup in these measurements is not meant as a specific real application scenario, but as an easy-to-setup way of measuring current consumption in a generic battery-powered ZED running Z-stack and CC2538, which is polling its parent and receiving messages and sending replies.

4.2.1 Programming

The following two sections describe how to setup, compile and download firmware to the ZC (Section 4.2.2) and the ZED node (Section 4.2.3). The default settings for the SampleLight project have power saving disabled; hence, the following instructions describe how to set up the project with power saving enabled on the ZED to allow it to go into sleep mode (PM2), which consumes approximately1.6 µA.

In order to be able to follow the detailed description below for the measurement setup, you need to have the correct software installed (Z-Stack and IAR ARM EW); for details see Section 3.2.1.

In the following, the boards (CC2538EM) that run the application are programmed by simply connecting them to a Smart06EB that is connected to the PC via a USB cable included in the kit. When connected in such a way, the debug feature in the IAR EW can be used to program the devices as the EW will automatically program the device with the current project open in the EW for debugging it (after compiling it, if needed). Then, you can simply stop the debugging process and the firmware will remain on the CC2538EM. For more details, see the *Z-Stack Home Sample Application User’s Guide* [4].

The CC2538EM programmed as an end device can then be mounted on the SmartRF06 battery board. Or, the SmartRF06EB board current measurement setup, as described in [11], can be used to measure current consumption.

For the measurements made in this application report, CC2538 with end device application was mounted on a SmartRF06BB.

4.2.2 ZC Node

Open the workspace file SampleSwitch.eww with the correct version of the IAR ARM EW. The project file is found in the following folder after installing the Z-Stack-Home-1.2.0 [3]:

C:\Texas Instruments\Z-Stack Home 1.2.0\Projects\zstack\HomeAutomation\SampleSwitch\CC2538\

Next, choose the CoordinatorEB configuration.

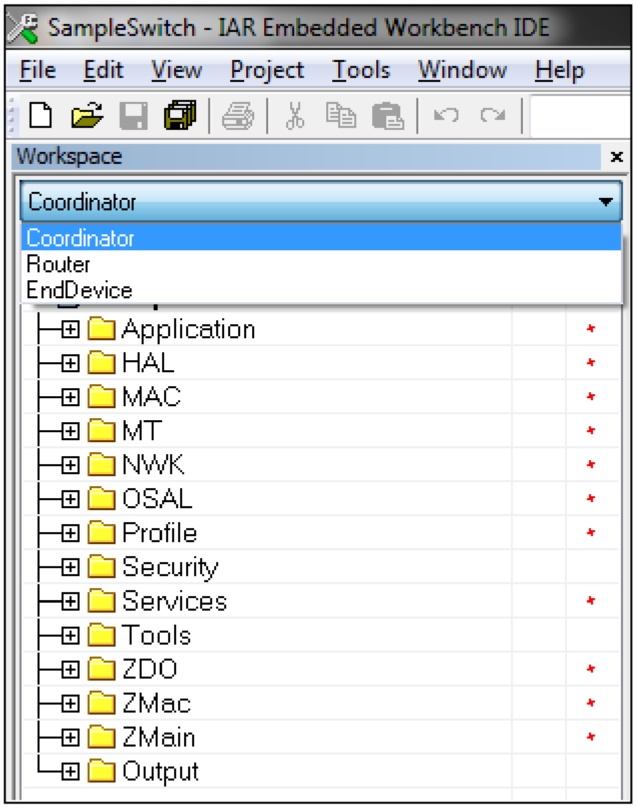

Figure 3. Choosing Workspace

In Section 4.2.4, it is explained how to establish the binding between the light and the switch. In order to enable the features to store and restore the binding information, you have to add the NV_RESTORE compile option to the project as shown in Figure 4. For more info about the NV_RESTORE compile option, see [13] and other Z-Stack documentation.

Always when programming a board with the NV_RESTORE option (that enables the node to save its current network state, bindings, and so forth, such that they are still there after a power toggle or failure on the board) for the first time with the code, it is important to erase the flash to ensure that all old data is erased to avoid that a freshly programmed node behaves strange based on old NV data. The erase flash option can be used first to perform the erase operation in the project options as shown in Figure 5.

As the final step, start the debug process *Project → Debug* (or simply CTRL+D). Now the CC2538EM board will be programmed with the SampleSwitch application and will act as ZC during the measurements. Simply stop the debugger (CTRL+SHIFT+D) and the CC2538EM is ready to be used. To start the ZC, simply toggle the power on the board.

For the measurements, the programmed CC2538 EM board was plugged into a SmartRF06EB board that is either powered via USB or batteries. For details, see [11].

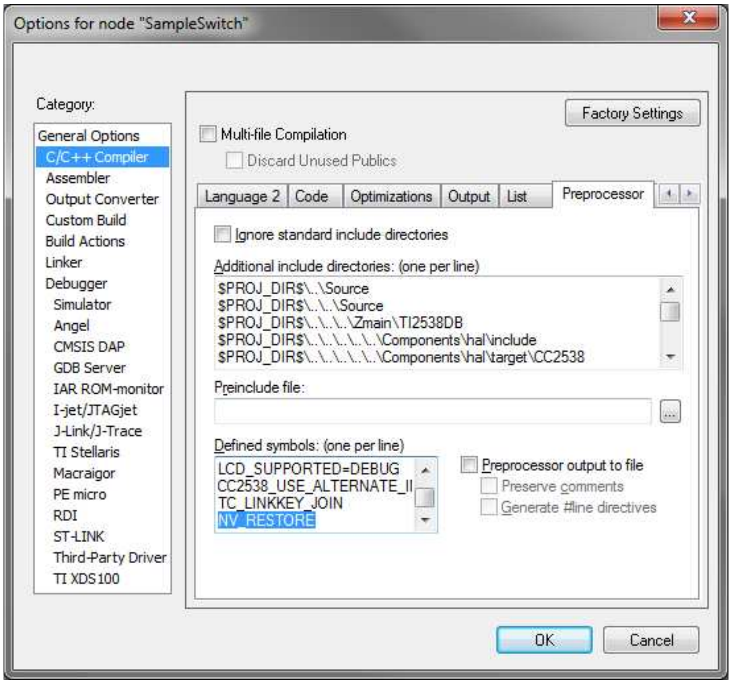

Figure 4. Setting NV_RESTORE Compile Option

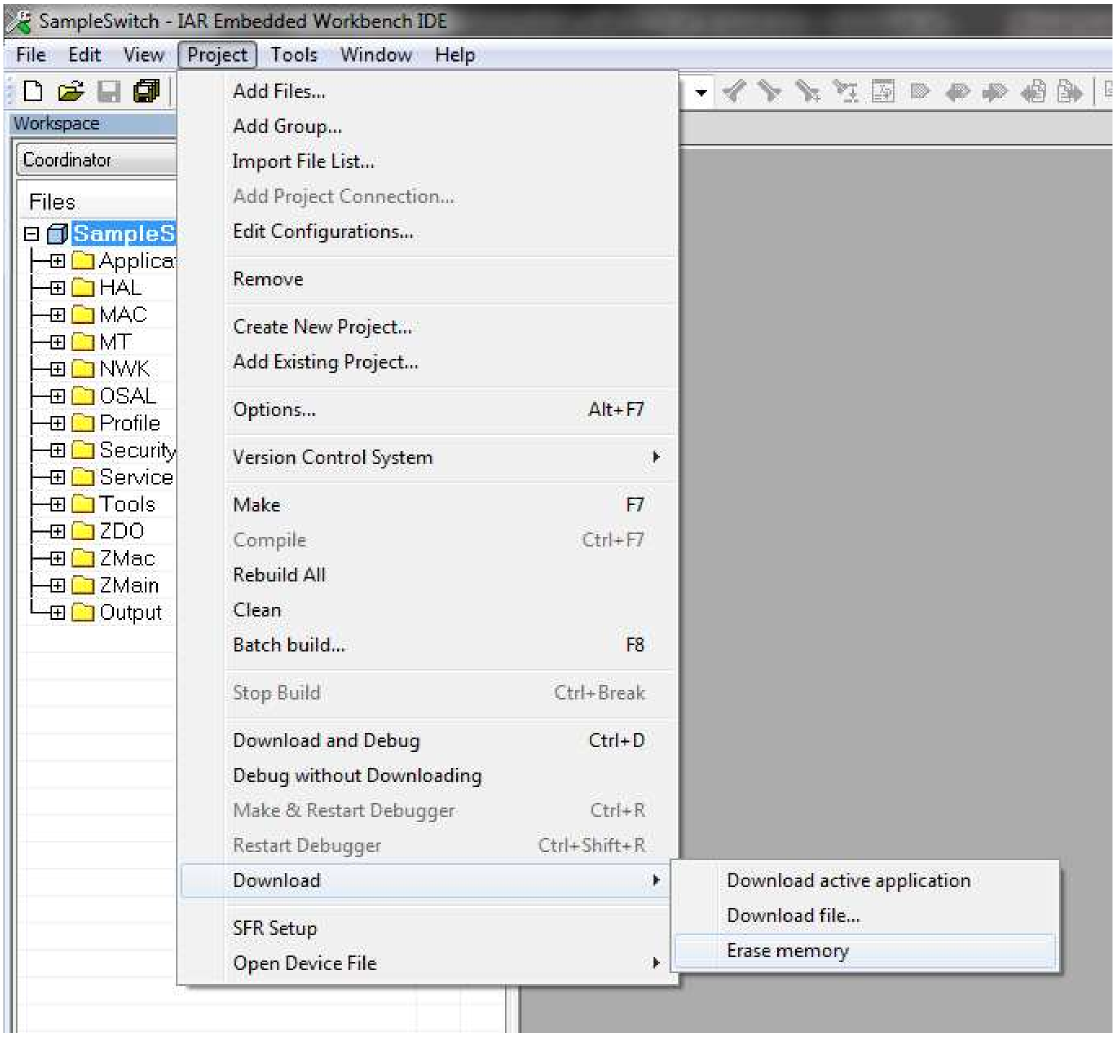

Figure 5. Erase Flash Option

4.2.3 ZED Node

Open the workspace file SampleLight.eww with the correct version of the IAR ARM EW. The project file is found in the following folder after installing the Z-Stack-Home-1.2.0 [3]:

C:\Texas Instruments\Z-Stack Home 1.2.0\Projects\zstack\HomeAutomation\SampleLight\CC2538

Next, choose the *EndDeviceEB-Pro* configuration.

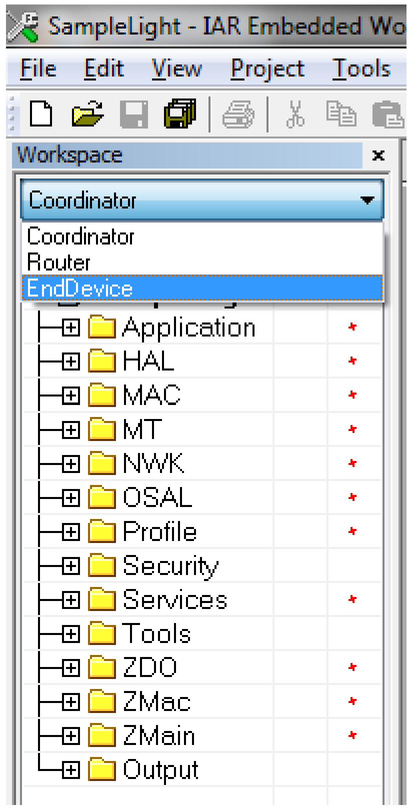

Figure 6. Choosing Workspace

The power saving feature (implemented in the Z-Stack for ZEDs) is not enabled by default (as it would hinder the debugging in the development phase). In order to enable the power saving feature for the measurement setup simply set the correct compile options described below by setting the Defined Symbols in the IAR EW. For all the details about the power saving functionality, see [8].

As already mentioned for the ZC Node it is also important to add the NV_RESTORE compile option. For more details, see Section 4.2.4, as it is explains how to establish the binding between the light and the switch.

Go to *Project* → *Options* (ALT+F7) and find the *C/C++ Compiler* → *Preprocessor* and make sure that the following values are included to enable network polling, power saving, and remembering the binding (see also):

- NWK_AUTO_POLL
- xLCD_SUPPORTED=DEBUG
- POWER_SAVING
- NV_RESTORE
- FEATURE_8MHZ_HYBRID_POWER_SAVING

Compile option FEATURE_8MHZ_HYBRID_POWER_SAVING is added to use the 8 MHz Clock on the CC2538. Reducing the Clock frequency may or may not be required as per application requirement. For the purpose of this application note, 8 MHz Clock frequency was used to reduce the current consumption for a sleepy end device (see Appendix A).

Additionally, the 8 MHz Clock frequency need to be setup by modifying the *SysCtrlClockStartSetting()* function in file hal_sys_ctrl.c.

```
void SysCtrlClockStartSetting(void)
{
   /* Setup the clock startup sequence to 32 MHz external
    * osc and 32k sourced from external oscillator
    */
  IOCPadConfigSet(GPIO_D_BASE, 0xC0, IOC_OVERRIDE_ANA);

  //SysCtrlClockSet(OSC_32KHZ, false, SYS_CTRL_SYSDIV_32MHZ); Comment Out
  SysCtrlClockSet(OSC_32KHZ, false, SYS_CTRL_SYSDIV_8MHZ);   // set CPU Clock Freq
                                                             // to 8MHz
}
```

It is important to note that when enabling these compile options the code changes included reconfigure the system clock frequency to 32 MHz before going to sleep. And after wake up, the clock frequency is set to 8 MHz again. This hybrid mode is used because the MAC timer's start and sync stop functionality requires system clock to be 32 MHz. You can also note this in the explanation of the power profile plots in Table 2 through Table 5, in section 5 and 6 of this application report.

Also remember for the ZED to erase the flash when programming as shown in Figure 5 for the ZC.

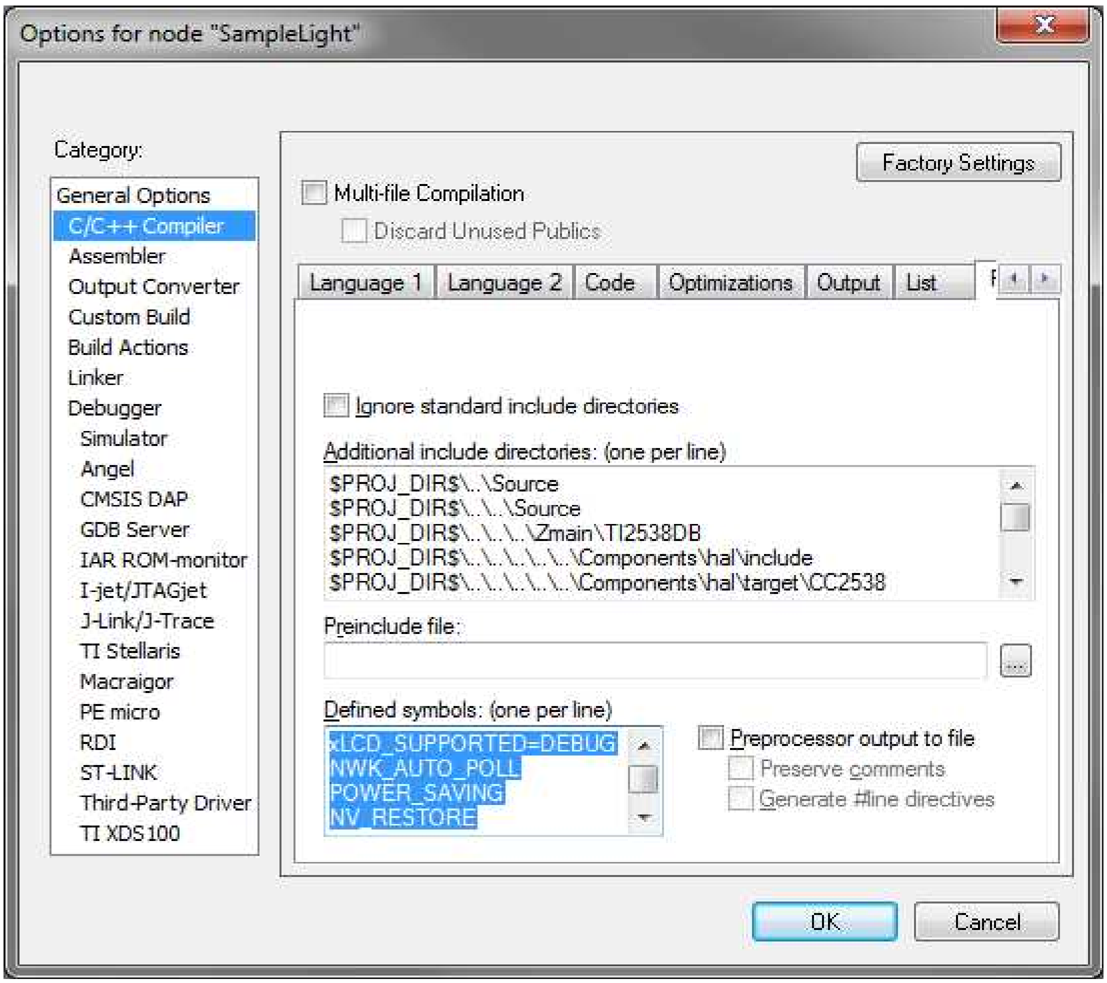

Figure 7. Setting Compile Options for the ZED

To make it easier to catch the measurements on the oscilloscope, it is useful to adjust the polling rate in the ZED, which determines how often the device wakes up from sleep (PM2) and sends a data request to its parent to poll for pending data (queued messages). A poll rate of 5 seconds (-DPOLL_RATE=5000) has been used as shown in Figure 8.

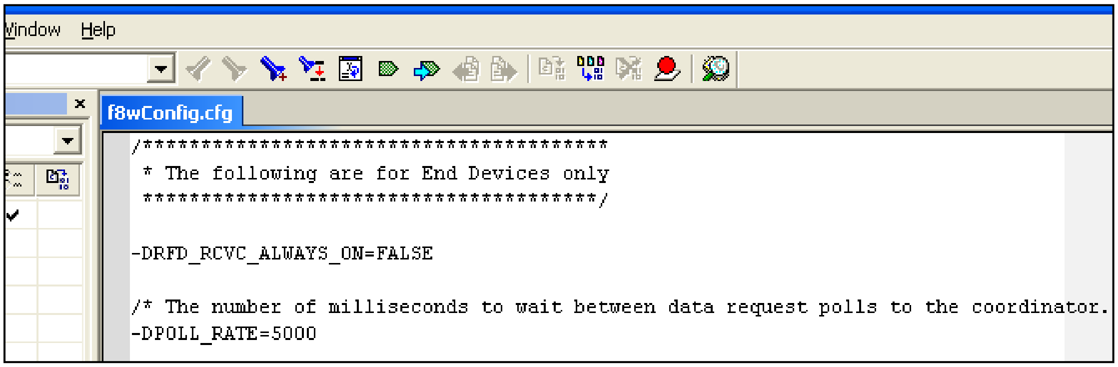

Figure 8. Setting Up Poll Rate

NOTE: In order to save power one might want to increase the polling rate even further; however, one should be aware that there are basically two limiting factors for the polling rate:

- The length of the timer that is used to implement this (max. 65 sec) and 2)
- In a ZigBee network, parents only keep the data buffered for their children for a certain time (hence, a ZED that sleeps too long would miss messages as they time out in its parent.

One could of course disable polling completely (if the device is not supposed to receive data); however, this might lead to high latency as the device (when waking up) first has to poll and if the parent is gone re-join or orphan join the network again before it can send its data. The latter can take a long time or become a problem if the network in the meantime changed channel or security key. For more details see the ZigBee specification.

The sample application is by default programmed to check for user input (for example, pressing buttons or using the joystick). This also periodically wakes up the MCU and consumes power. In order to turn this off as it is too application board specific and, hence, not relevant for this application report, you need to turn off the feature called *key polling*. This is done by changing the following line in the *InitBoard(byte level)* function that can be found in the *Onboard.c* file.

From: `OnboardKeyIntEnable = HAL_KEY_INTERRUPT_DISABLE;`

To: `OnboardKeyIntEnable = HAL_KEY_INTERRUPT_ENABLE;`

```
    /* Initialize Key stuff */
    OnboardKeyIntEnable = HAL_KEY_INTERRUPT_ENABLE; //HAL_KEY_INTERRUPT_DISABLE;
    HalKeyConfig( OnboardKeyIntEnable, OnBoard_KeyCallback);
  }
```

Turn off the LED to measure only the consumption by the CC2538 End device. Change HAL_LED to False in the file hal_board_cfg.h.

```
/* Set to TRUE enable LED usage, FALSE disable it */
#ifndef HAL_LED
#define HAL_LED FALSE // Set to FALSE
#endif
#if (!defined BLINK_LEDS) && (HAL_LED == TRUE)
#define BLINK_LEDS
#endif
```

In out-of-box Sample application, all the peripherals are turned on. In order to reduce power consumption, the peripherals that are not being used by the application can be turned off. In the *SysCtrlRunSetting()* function in *hal_sys_ctrl.c* file, comment the lines that Enable the un-used peripherals. For the purpose of this application report, changes as shown below were made.

```
void SysCtrlRunSetting(void)
{

//Unused Peripherals Not Turned On to reduce Power Consumption

  /* Enable General Purpose Timers 0, 1, 2, 3 when running */
//  SysCtrlPeripheralEnable(SYS_CTRL_PERIPH_GPT0);
//  SysCtrlPeripheralEnable(SYS_CTRL_PERIPH_GPT1);
//  SysCtrlPeripheralEnable(SYS_CTRL_PERIPH_GPT2);
//  SysCtrlPeripheralEnable(SYS_CTRL_PERIPH_GPT3);

  /* Enable SSI 0, 1 when running */
//  SysCtrlPeripheralEnable(SYS_CTRL_PERIPH_SSI0);
//  SysCtrlPeripheralEnable(SYS_CTRL_PERIPH_SSI1);

  /* Enable UART 0, 1 when running */
//  SysCtrlPeripheralEnable(SYS_CTRL_PERIPH_UART0);
//  SysCtrlPeripheralEnable(SYS_CTRL_PERIPH_UART1);

  SysCtrlPeripheralReset(SYS_CTRL_PERIPH_AES);
//  SysCtrlPeripheralReset(SYS_CTRL_PERIPH_PKA);


  /* Enable I2C, AES and PKA running */
//  SysCtrlPeripheralEnable(SYS_CTRL_PERIPH_I2C);
//  SysCtrlPeripheralEnable(SYS_CTRL_PERIPH_PKA);
  SysCtrlPeripheralEnable(SYS_CTRL_PERIPH_AES);

  /*
   * Enable RFC during run. Please note that this setting is
   * only valid for PG2.0. For PG1.0 since the RFC is always on,
   * this is only a dummy  instruction
   */
  SysCtrlPeripheralEnable(SYS_CTRL_PERIPH_RFC);

}
```

As the final step simply start the debug process *Project → Debug* (or simply CTRL+D). Now the CC2538 EM board will be programmed with the SampleLight application with power savings enabled and will act as ZED during the measurements.

Stop the debugger (CTRL+SHIFT+D) and follow the instructions in Section 4.2.4 to establish the correct binding, before detaching the CC2538EM from the SmartRF06EB board (that was used to program it and establish the binding) and connect it to the wiring as shown in Figure 2.

4.2.4 Binding

After following the above steps, the freshly programmed CC2538EMs ZC was mounted on SmartRF06EB and the ZED was mounted on SmartRF06BB to perform the binding.

In a first step, you should turn on the ZC to start the network. The LCD display on the EB (to which the ZC is attached) should indicate when the network is established by displaying “ZigBee Coord” and “Newtwork ID: <PANID>. As soon as the network is started, you can turn on the ZED and the ZED will then join the ZC. You can observe the OTA packets and device joining the coordinator using a packet sniffer.

As soon as the network is established, you should press the Button 2 on both boards (on the right, seen from the lower side of the board when it is oriented such that you can read the silk screen that says “RIGHT”). Then the ZED is bound to the switch. For details, see the *Home Automation Sample Application* description in [4], Section 3.2.

Packet Sniffer was used to verify the exchange of the commands and the success of the binding by pressing Button UP on the coordinator and observing that the coordinator sends a Toogle ON/OFF command. The end device receives this command when it does a data poll to retrieve this message. And the end device then sends a Default response command to the toggle ON/OFF command.

From now on, you can send the toggle command to the ZED by pressing the up button on the SmartRF06EB or the SmartRF06 battery board on which coordinator is mounted.

For more details, see the Home Automation sample application description in [4]. At this stage, the ZED can be turned off and taken off from the SmartRF06 board it was plugged into in order to connect it to the SmartRF06BB as shown in Figure 2.

Also the ZC can now be turned off (just like the ZED) as both were compiled with the NV_RESTORE compile options, which causes the established binding, PAN ID, and short address (next to other parameters) to be saved.

From now on, you can always turn them on again and the network will form itself again (the ZED performs an orphan join to the ZC). From that point, the ZED starts again to poll the ZC for data. If the joystick on the ZC board is pressed up, the toggle command is picked up by the ZED.

The operation where the ZED polls its parent (the ZC) periodically to see whether there are pending messages for it, is in the following referred to as:

- Operation1 when there is no data pending
- Operation2 when there is a message (“toggle the light”) from the switch (ZC) pending, received, processed, and acknowledged by the light application (ZED).

4.3 Calculation of the Average Current Consumption

The calculation of the current is based on the well known relation (Ohm’s Law):

$V = I * R$

where, V, I, and R represent the voltage, the current, and the resistance.

By measuring at the power supply side, the test system observes the current consumed by the CC2538EM. The *Current Measurement* section in the SmartRF06EB User Guide [11] also describes a method to measure the current consumption while the CC2538EM is plugged into the SmartRF06EB.

As explained in [12], the value of R should not be too large, since it reduces the effective voltage over the evaluation module (EM) itself:

$$V_{TARGET\ BOARD} = V_{POWER\ SUPPLY} - I * R \quad (1)$$

In order to keep the measurement system simple and easy to reproduce, accepting the error introduced by the resistor, the value was chosen to be 2.8 Ω.

5 Measurements

This section shows and explains the results that were obtained with the setup explained in the previous sections. In the measurement setups, the TX packets are sent out at 0-dBm power. Since 2.8-Ω resistor is used, 28 mV in the measurement is translated into 1 mA current consumption.

5.1 Measurements for Operation1

In Operation1, the ZED polls the ZC, sending MAC Data Req command, and it is assumed that there is MAC ACK from the ZC with Frame Pending subfield of 0, which means the ZC does not have any message to send to the ZED. Therefore, the ZED does not need to wait for a message and can go to sleep right away.

MAC Data Req packet is 18 bytes long including 6-byte PHY overhead, which takes 576 µs (18 byte x 8 bit/byte x 4 µs/bit) over-the-air.

In the same manner, MAC ACK is 11 byte-long and takes 352 µs over-the-air. MAC Data Req and MAC ACK are plotted at Point are potted at Point 7-8 and 10-11 in Figure 9, respectively.

The time period length of each unit operation section is constant except the CSMA/CA (Point 3-4) time, which depends on the channel condition. 1.2025 ms for the unit operation CSMA/CA in Table 2 is the value averaged from 20 measurement samples.

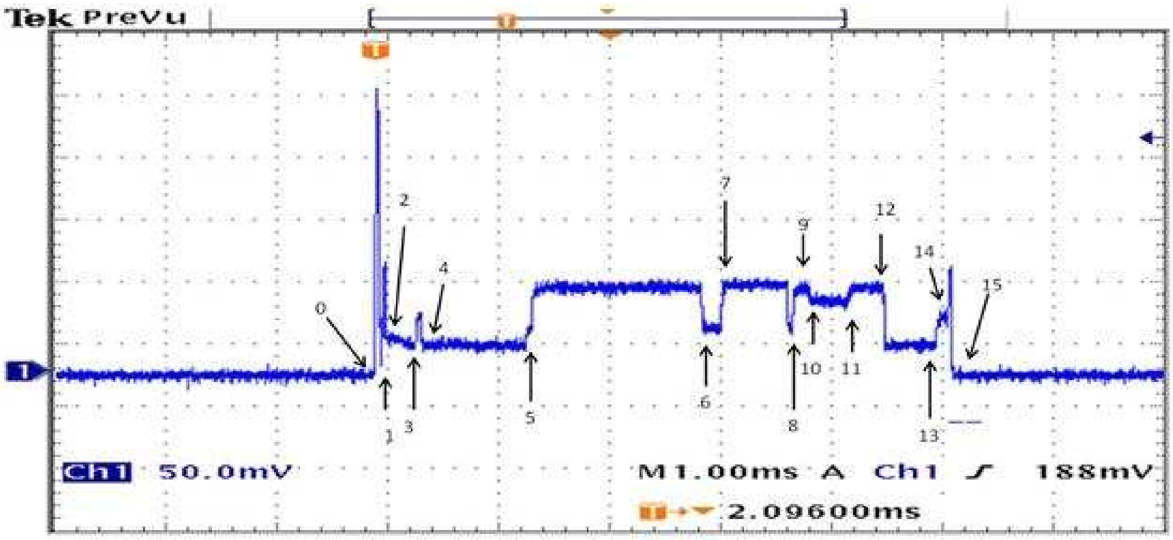

Figure 9. Current Consumption Measurement Plot on Operation1

Table 2. Current Consumption Measurement Breakdown on Operation1

Section	State Description	Voltage (mV)	Current (mA)	Time (ms)	Power (mA*ms)
Before 0	Power Mode 2	0	0.0016	0	0
Point 0 to 1	Wake up from Sleep Peak1	268	95.71	0.06	7.295
Point 1 to 2	Wake Up From Sleep Peak 2	90	32.14	0.048	1.54
Point 2 to 3	MCU Wake Up from Sleep 32 MHz Clock	28	10	0.232	2.32
Point 3 to 4	MCU Clock change to 8 MHz	44	15.71	0.064	1.005
Point 4 to 5	MCU running on 8 MHz	21	7.5	0.876	6.57
Point 5 to 6	CSMA-CA Before sending the Mac Data Req	68	24.28	1.063	25.83
Point 6 to 7	Switch from RX to TX	36	12.85	0.192	2.46
Point 7 to 8	Packet TX (Mac Data Req)	70	25	0.576	14.4
Point 8 to 9	Switch from TX to RX	36	12.85	0.048	0.61
Point 9 to 10	Switch form TX to RX	64	22.85	0.152	3.47
Point 10 to 11	Radio Receiving the MAC ACK	58	20.71	0.348	7.20
Point 11 to 12	Radio/Code Processing	66	23.57	0.088	2.07
Point 12 to 13	MCU in active mode running on 8MHz	22	7.85	0.924	7.26
Point 13 to 14	CPU Clock Change to 32MHz	46	16.42	0.181	2.98
Point 14 to 15	Processing and shut down	83	29.64	0.048	1.42
After 15	Power Mode 2	0	0.0016	0	0
Total				**4.9**	**86.48**

5.2 *Measurements for Operation2*

In Operation2, the ZED polls the ZC, sending MAC Data Req command, and the ZC responds to the ZED with MAC ACK where Frame Pending subfield is 1, which means the ZC has a message heading to the ZED. Upon the reception of the MAC ACK with Frame Pending subfield of 1, the ZED waits for a message (the Toggle command in this example) that the ZC is sending soon. After receiving the message, the ZED responds to the ZC sending the Default Response command.

The Toggle command packet is 54 bytes long (including the 6 bytes PHY header), which takes 1.728 ms, and the Default Response command packet is 58 bytes long, which takes 1.856 ms. The Receiving Toggle command and the sending Default Response command are plotted at Point 12 to 13 and Point 18 to 19 in Figure 10, respectively.

The Time period length of each unit operation section is constant except Point 4 to 5, Point 5 to 6, Point 11 to 12, Point 16 to 17, and Point 25 to 26, which depend on the channel condition. The time values for those three unit operation sections in Table 3 are the averaged values from 20 measurement samples.

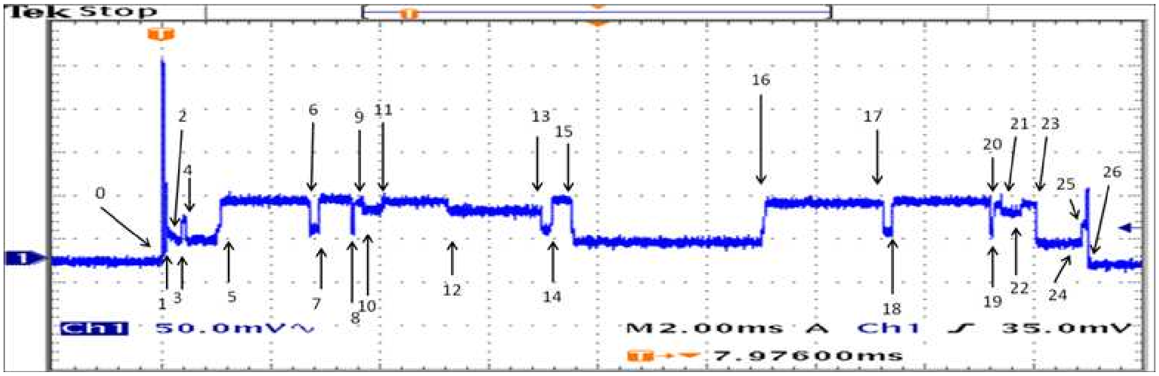

Figure 10. Current Consumption Measurement Plot on Operation2

Table 3. Current Consumption Measurement Breakdown on Operation2

Section	Unit Operation Description	Voltage (mV)	Current (mA)	Time (ms)	Consumption (mA*ms)
Before 0	Power Mode 2		0.0016		
Point 0 to 1	Wake Up From Sleep Peak 1	268	95.71	0.06	5.74
Point 1 to 2	Wake Up From Sleep Peak 2	90	32.14	0.048	1.54
Point 2 to 3	MCU Wake Up from Sleep 32 MHz Clock	28	10	0.232	2.32
Point 3 to 4	MCU Clock Change to 8 Mhz	44	15.71	0.064	1.00
Point 4 to 5	MCU running on 8 MHz clock	25	8.928	0.841	7.50
Point 5 to 6	CSMA-CA (before sending the MAC Data Req)	69	24.64	1.104	27.20
Point 6 to 7	Switch from RX to TX	39	13.92	0.192	2.67
Point 7 to 8	Transmit MAC Data Request	69	24.64	0.576	14.19
Point 8 to 9	Switch from TX to RX	39	13.92	0.064	0.89
Point 9 to 10	Switch from TX to RX	65	23.21	0.144	3.34
Point 10 to 11	Receiving MAC ACK from Coordinator	58	20.71	0.36	7.45
Point 11 to 12	Radio in RX mode (processing MAC ACK and then waiting for the packet)	68	24.28	1.334	32.3
Point 12 to 13	Receiving Toggle command	59	21.07	1.72	36.24
Point 13 to 14	Switch from RX to TX	38	13.57	0.176	2.38
Point 14 to 15	Transmit MAC ACK	69	24.64	0.384	9.46
Point 15 to 16	CPU on 8 MHz Clock Processing the Toggle Command	22	7.857	4.04	31.74
Point 16 to 17	CSMA-CA	67	23.92	0.882	21.10
Point 17 to 18	Switch from RX to TX	36	12.85	0.216	2.77
Point 18 to 19	Transmit the ON OFF Default Response	70	25	1.77	44.25
Point 19 to 20	Switch from TX to RX	38	13.57	0.064	0.86
Point 20 to 21	Switch from TX to RX	67	23.92	0.136	3.25
Point 21 to 22	RX MAC ACK	60	21.42	0.368	7.88
Point 22 to 23	RX Processing	66	23.57	0.104	2.45
Point 23 to 24	CPU on 8 MHz Clock	21	7.5	1.24	9.3
Point 24 to 25	MCU Clock Change to 32 MHz	46	16.42	0.22	3.61
Point 25 to 26	Processing and Shut down	87	31.07	0.048	1.49
After 26	Sleep Mode 2		0.0016		
				16.387	**283.11**

6 Application to a Practical Use Case

From Section 4 through Section 5, ZC Switch and ZED Light model has been used for ease of measurement setup excluding any possible CC2538-external current consumption. However, the typical and realistic use case is a combination of a Switch as a battery-powered sleeping ZED and a Light as a mains-powered ZC or ZR. With this configuration, it is assumed that the ZED Switch sends the Toggle command and polls for the receiving Default Response from the ZC Light. The message exchange sequence can be seen in Figure 11, also described below:

- The End device sends an On and Off Cluster toggle command, and receives a MAC ACK with message pending bit set to zero from the coordinator.
- Later when the end device polls the coordinator, the coordinator sends a MAC ACK with pending bit set to 1 suggesting it will send a buffered message. This is the Default response to the Toggle command sent earlier.
- The end device then sends a MAC ACK to the received Default response.
- The end device then wakes up again after the QUEUED_POLL_RATE to check whether or not there are any more messages buffered at the parent. It gets a MAC ACK with pending bit set to zero. And the end device then enters the PM2.

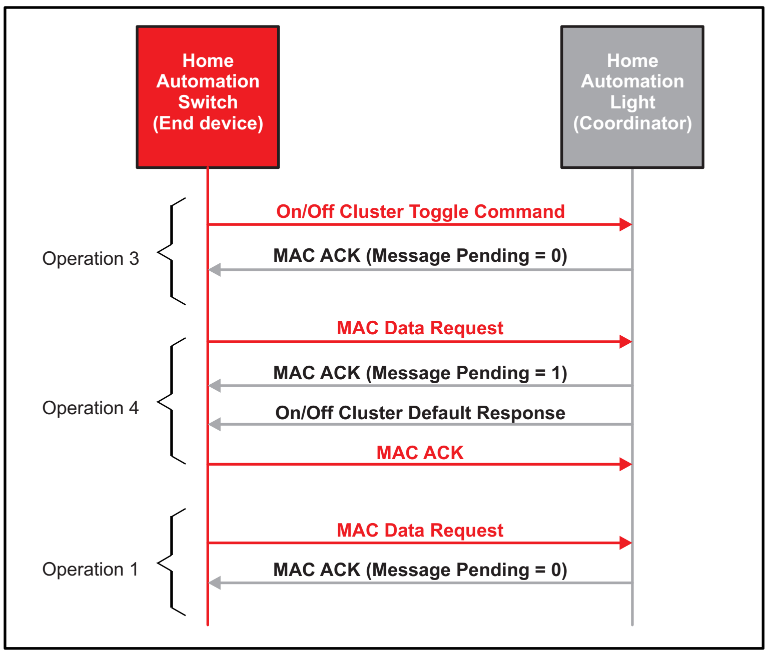

Figure 11. Message Sequence: End Device Switch Sends Toggle Command to Light as a Coordinator Zigbee Logical Device

Though the measurement on that configuration has not been done in this document, it can be calculated from the unit operation-wise results you got in Section 5. Current consumption for Operation 3 and Operation 4 shown in Figure 11 are described in Section 6.1.

6.1 *Power Consumption Estimation per Operation*

6.1.1 Operation3 - Toggle Command TX

Transmission of Toggle command has the same timing pattern as Table 2 except the unit operation of Point 5-6. The length of Point 7 to 8 can be calculated theoretically (based on 250 Kbps PHY transmission rate for ZigBee at 2.4GHz). Consequently, the current consumption estimation of Operation3 will be as shown in Table 3.

Table 4. Current Consumption Estimation Breakdown on Operation3

Section	State Description	Voltage (mV)	Current (mA)	Time (ms)	Power (mA*ms)
Before 0	Power Mode 2	0	0.0016	0	0
Point 0 to 1	Peak 1	268	95.71	0.06	5.74
Point 1 to 2	Peak 2	90	32.14	0.048	1.54
Point 2 to 3	MCU Wake Up from Sleep 32 MHz Clock	28	10	0.232	2.32
Point 3 to 4	MCU Clock Change to 8 Mhz	44	15.71	0.064	1.00
Point 4 to 5	MCU running on 8 MHz	22	7.857	0.876	6.88
Point 5 to 6	CSMA	68	24.28	1.063	25.83
Point 6 to 7	Switch from RX to TX	34	12.14	0.192	2.33
Point 7 to 8	Packet TX - Send the Toggle Command (Time calculated theoretically based on 54 Byte packet)	68	24.28	1.728	41.96
Point 8 to 9	Switch from TX to RX	34	12.14	0.192	2.33
Point 9 to 10	Reception of MAC Acknowledgment from Coordinator	64	22.85	0.152	3.47
Point 10 to 11	Radio remaining in RX mode and processing the MAC ACK	56	20	0.364	7.28
Point 11 to 12	Radio/Code Processing	64	22.85	0.308	7.04
Point 12 to 13	MCU in active mode running on 8 MHz	22	7.85	0.465	3.65
Point 13 to 14	MCU Clock Change to 32 MHz	41	14.642	0.181	2.66
Point 14 to 15	Processing and Shut down	82	29.2859	0.052	1.52
After 15	Power Mode 2		0.0016		
Total				**5.97**	**115.59**

6.1.2 Operation4 - Polling Followed by Default Response RX

Operations of polling and receiving Default Response command have the same timing pattern as Point 0-12 of Table 3 except the unit operation of Point 11 to 12. can be calculated theoretically (based on 250 Kbps PHY transmission rate for ZigBee at 2.4GHz). In addition to those, Operation4 will have the unit operation of processing and shutdown as Point 25 to 26 of Table 3. Consequently, the current consumption estimation of Operation4 will be as shown in Table 5.

Table 5. Current Consumption Estimation Breakdown on Operation4

Section	Unit Operation Description	Voltage (mV)	Current (mA)	Time (ms)	Consumption (mA*ms)
Before 0	Power Mode 2		0.0016		
Point 0 to 1	Peak 1	268	95.714	0.06	5.7428
Point 1 to 2	peak 2	90	32.142	0.048	1.5428
Point 2 to 3	MCU Wake Up From Sleep 32 MHz Clock	28	10	0.232	2.32
Point 3 to 4	MCU Clock Change to 8 Mhz	44	15.714	0.064	1.0057
Point 4 to 5	MCU running on 8 MHz clock	22	7.8571	0.841	6.6078
Point 5 to 6	CMSA/CA algorithm. Radio in RX mode	63	22.5	1.104	24.84
Point 6 to 7	Switch from RX to TX	32	11.428	0.192	2.1942
Point 7 to 8	Transmitting MAC Data Request. Radio in TX mode	65	23.214	0.58	13.464
Point 8 to 9	Switch from TX to RX	32	11.428	0.2	2.2857
Point 9 to 10	Receiving MAC ACK from Coordinator	58	20.714	0.48	9.9428
Point 10 to 11	Radio in RX mode (processing MAC ACK and then waiting for the packet)	65	23.214	1.334	30.967
Point 11 to 12	Receiving Default Response command (Theoretical time calculation based on 58 Byte default response packet)	54	19.285	1.856	35.794
Point 13 to 14	Switch from RX to TX	32	11.428	0.192	2.1942
Point 14 to 15	Transmitting MAC Acknowledgment. Radio in TX mode	65	23.214	0.384	8.914
Point 15 to 16	MCU running on 8 MHz clock	17	6.0714	0.816	4.9542
Point 16 to 17	MCU Clock Change to 32 Mhz	40	14.285	0.22	3.1428
Point 17 to 18	Processing and Shut down	79	28.214	0.048	1.3542
After 18	Power Mode 2		0.0016		
Total				**8.651**	**157.2686**

6.2 Estimation for Usage Scenario

In this section, you will estimate per-day current consumption amounts based on daily usage scenarios and calculate the battery life for each scenario. It is assumed that 1 AA battery whose capacity is 3000 mAh is used.

For ease of calculation, defined here is the constant CC_{SLEEP}, per-day charge consumption in sleep (PM2) mode. Since the switch is supposed to sleep most of time and the time period where it is awake is relatively negligible, assume the light is consuming sleep(PM2) current all the time. CC_{SLEEP} is calculated as shown in Equation 2.

$$CC_{SLEEP} = 0.0016mA * 1000 ms * 60 * 60 * 24 = 138240 \tag{2}$$

Consider two example usage scenarios to present how to estimate per-day charge consumption in various actual usages.

- Usage Scenario 1

  In Scenario 1, assume the switch polls the light every 5 seconds and toggles the light 20 times a day. Assuming there is no communication failure, this scenario has (60 x 60 x 24 / 5 – 20) = 17260 times of Operation1, 20 times of Operation3, 20 times of Operation4 and 20 more times of Operation1.

  Therefore, the total per-day charge consumption CC_{TOTAL} is calculated as:

$$\begin{aligned}
CC_{Total} &= CC_{SLEEP} + CC_{Operation1} * 17260 + CC_{Operation3} * 20 + CC_{Operation4} * 20 + CC_{Operation1} * 20 \\
CC_{Total} &= 138240 + (86.48 * 17260) + (115.59 * 20) + (157.61 * 20) + (86.48 * 20) \\
CC_{Total} &= 1638150\ mA * ms \\
CC_{Total} &= 0.455\ mAh
\end{aligned} \tag{3}$$

  And, the battery life can be calculated as shown in Equation 4.

$$Battery\ Life = \frac{3000}{0.449} = 6593.4\ days = \sim 18.06\ years \tag{4}$$

- Usage Scenario 2

  In Scenario 2, assume the switch polls the light every second and toggles the light 10 times a day. Assuming there is no communication failure, this scenario has (60 x 60 x 24 / 1 – 10) = 86390 times of Operation1, 10 times of Operation3, and 10 times of Operation4. Therefore, the total per-day charge consumption CC_{TOTAL} is calculated as shown in Equation 5.

$$\begin{aligned}
CC_{Total} &= CC_{SLEEP} + CC_{Operation1} * 86390 + CC_{Operation3} * 10 + CC_{Operation4} * 10 + CC_{Operation1} * 10 \\
CC_{Total} &= 138240 + (86.48 * 86390) + (115.59 * 10) + (157.26 * 10) + (86.48 * 10) \\
CC_{Total} &= 7613231\ mA * ms \\
CC_{Total} &= 2.115\ mAh
\end{aligned} \tag{5}$$

  And, the battery life can be calculated as shown in Equation 6.

$$Battery\ Life = \frac{3000}{2.115} = 1418.44\ days = \sim 3.88\ years \tag{6}$$

For more usage scenarios and corresponding results using the 'Usage Scenario' sheet of the accompanying spread sheet, see http://www.ti.com/lit/zip/swra456.

7 Conclusion

In this document, you have gone through the basics of performing current consumption measurements using the Z-Stack and the CC2538. Current consumptions were measured on two cases to get each unit operation's current consumption and applied the result to practical usage scenarios.

You can make your own measurement setup and take measurements. Also, you can calculate daily current consumption based on scenarios and finally estimate the battery life. All of these calculation and estimation formulas are provided in an associated spread sheet that can be downloaded from the following URL: http://www.ti.com/lit/zip/swra456.

8 References

1. *A Powerful System-On-Chip for 2.4-GHz IEEE 802.15.4, 6LoWPAN and ZigBee Applications Data Sheet* (SWRS096)
2. CC2538 Product Page: http://www.ti.com/product/cc2538
3. TI's ZigBee Stack – Z-Stack: http://www.ti.com/tool/z-stack
4. *Z-Stack Home Sample Application User's Guide* (Located at C:\Texas Instruments\ZStack-Home-1.2.0\Documents after installing the Z-Stack Home-1.2.0)
5. TI's ZigBee RF4CE Stack – RemoTI: (http://www.ti.com/tool/remoti)
6. *RemoTITM Power Consumption* (SWRA263)
7. *CC2538 SoC for 2.4-GHz IEEE 802.15.4 and ZigBee/ZigBee IP Applications User's Guide* (SWRU319)
8. Power Management For The CC2538.pdf (Located at C:\Texas Instruments\ZStack-Home-1.2.0\Documents after installing the Z-Stack Home-1.2.0)
9. CC2538 Development Kit: (http://www.ti.com/tool/cc2538dk)
10. Texas Instruments Packet Sniffer: (http://www.ti.com/tool/packet-sniffer)
11. *SmartRF06 Evaluation Board User's Guide* (SWRU321)
12. *Measuring the Power Consumption on eZ430-RF2480* (SWRA177)
13. Z-Stack Developer's Guide.pdf (Located at C:\Texas Instruments\ZStack-Home-1.2.0\Documents after installing the Z-Stack Home-1.2.0)
14. CC2538 Foundation Firmware
15. *Measuring Power Consumption of CC2530 With Z-Stack* (SWRA292)

Appendix A Current Consumption Measurement for Operation 1

This section provides the current consumption measurement performed for operation 1 (sending MAC Data Poll to the parent when parent has no data buffered) when the MCU is running at 32 MHz. As can be seen from the overall charge consumption in Table 6 vs the Table 2 results, the power consumption is reduced by using 8 MHz clock.

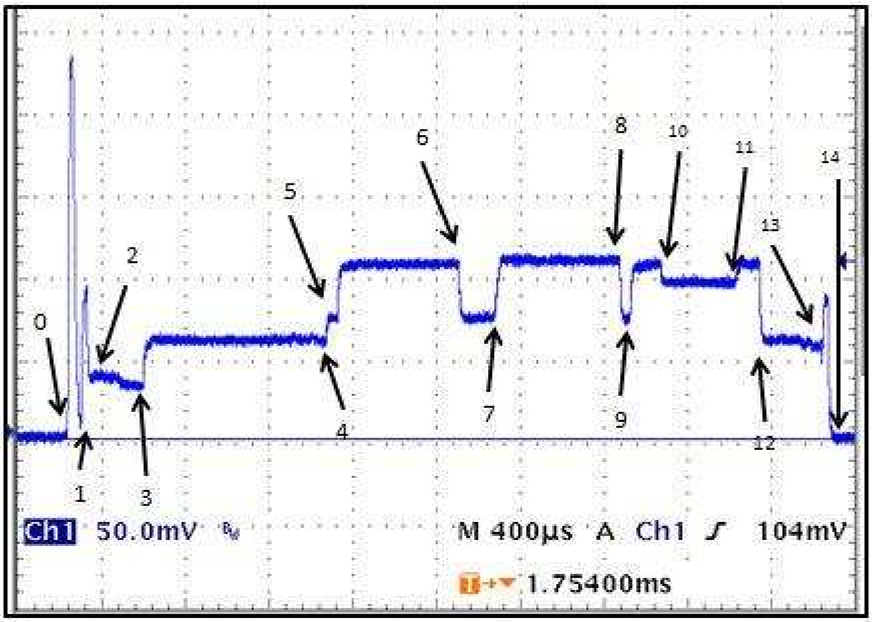

Figure 12. Current Consumption Measurement Breakdown on Operation1 With MCU at 32 MHz

Table 6. Current Consumption Measurement Breakdown on Operation1 With MCU at 32 MHz

Section	Unit Operation Description	Voltage (mV)	Current (mA)	Time (ms)	Consumption
Before 0	Power Mode 2	0			
Point 0 to 1	Power mode start up sequence.	230	82.14	0.064	5.25
Point 1 to 2	Power mode start up sequence.	86	30.71	0.044	1.35
Point 2 to 3	MCU active	36	12.85	0.262	3.36
Point 3 to 4	MCU running on 32MHz Clock	56	20	0.5	10
Point 4 to 5	MCU running on 32MHz Clock	70	25	0.06	1.5
Point 5 to 6	CSMA-CA before sending the MAC Data Request	102	36.42	0.584	21.27
Point 6 to 7	Switch from RX to TX	70	25	0.184	4.6
Point 7 to 8	Send the MAC Data Request	104	37.14	0.588	21.84
Point 8 to 9	Switch from TX to RX	70	25	0.048	1.2
Point 9 to 10	Reception of MAC ACK	101	36.07	0.156	5.62
Point 10 to 11	Radio in RX mode and processing the MAC ACK	91	32.5	0.356	11.57
Point 11 to 12	Radio code processing	101	36.07	0.112	4.04
Point 12 to 13	Processing before entring the sleep mode	57	20.35	0.292	5.94
Point 13 to 14	Processing before entring the sleep mode	80	28.57	0.04	1.14
Total				**3.29**	**98.71**

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as "components") are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI's terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers' products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers' products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI's goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or "enhanced plastic" are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265