# Lab 3A: Compile Switch On/Off and Enable Debug

This hands-on exercise will demonstrate how to use the Z-Wave Embedded SDK to enable serial debugging and compile a sample application using Simplicity Studio.

This exercise is part of the series “Z-Wave 1-Day Course”.

1: Include using SmartStart

2: Decrypt Z-Wave RF Frames using the Zniffer

**3A: Compile Switch On/Off and Enable Debug**

3B: Modify Switch On/Off

4: Understand FLiRS devices

**KEY FEATURES**

- Compile a Z-Wave Sample Application
- Enable serial debug
- Running and using a Z-Wave Sample Application
- Introduction to Z-Wave Command Classes

# 1 Introduction

This is the first exercise in the “Z-Wave 1-Day Course” series, where you will use Simplicity Studio to compile a Z-Wave sample application.

This exercise will guide you in how to set the appropriate frequency for your region and enable serial debug to facility serial output messages useful when testing a sample application.

## 1.1 Hardware Requirements

- 1 WSTK Main Development Board
- 1 Z-Wave Radio Development Board: ZGM130S SiP Module
- 1 UZB Controller
- 1 USB Zniffer

## 1.2 Software Requirements

- Simplicity Studio v4
- Z-Wave 7 SDK
- Z-Wave PC Controller
- Z-Wave Zniffer


**Figure 1: Main development board with Z-Wave SiP module**

## 1.3 Prerequisites

Previous Hands-On exercises has covered how to use the PC Controller and Zniffer application to build a Z-Wave network and capturing the RF communication for development purpose.

This exercise assumes you are familiar with these tools.

# 2 Compile your first Z-Wave Sample Application

## 2.1 Open Sample Project

1. Connect your Z-Wave hardware to the USB port of the computer and it should show up in the “Debug Adapters” section in Simplicity Studio.

2. Click once on the “J-Link Silicon Labs” which instructs to studio the show relevant information about Z-Wave 700.

3. Under “Software Example” click on the Switch On/Off sample application.

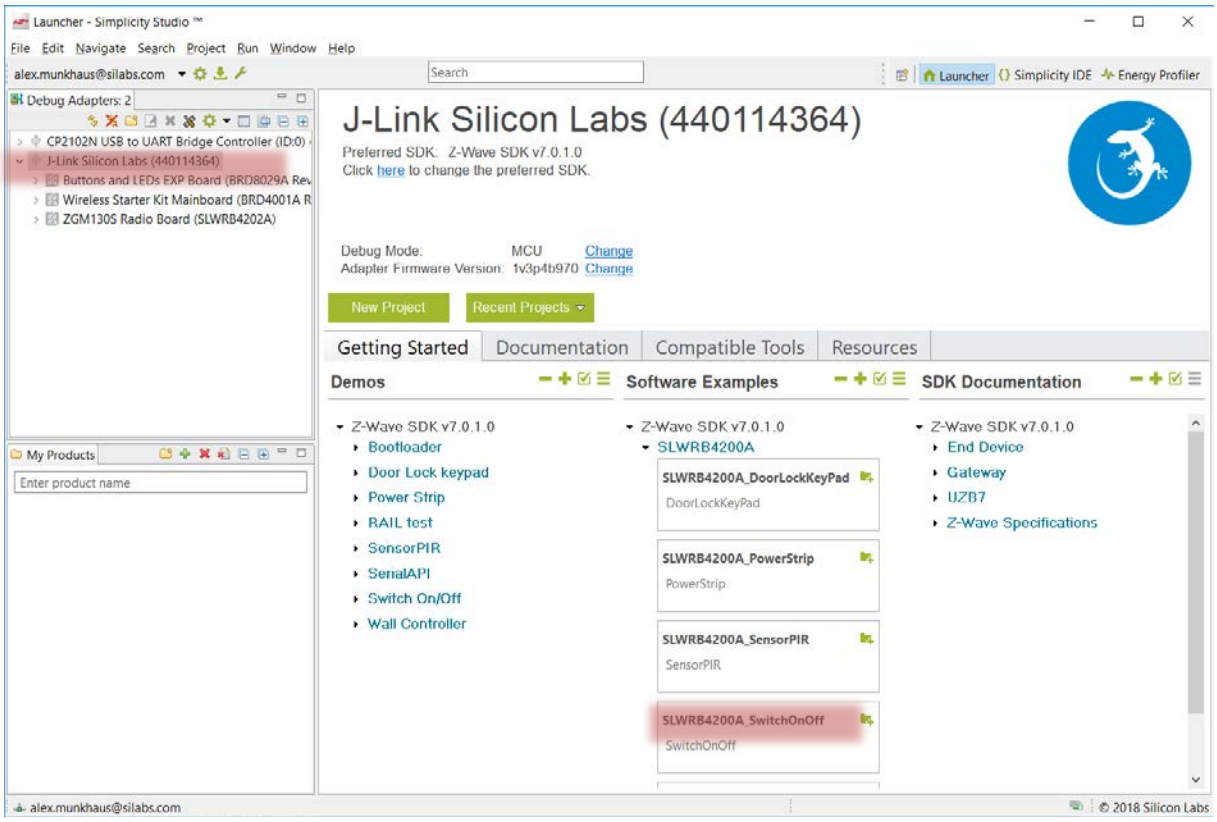


**Figure 2: Open a Z-Wave Sample Application: Switch On/Off**

## 2.2 Set the Frequency

The sample app will not compile just yet. You need to set the frequency that matches the region you intend to use the Z-Wave Product in.

- In the main source file “`SwitchOnOff.c`”, locate the variable `APP_FREQ:`

```
static const SRadioConfig_t RadioConfig =
{
  .iListenBeforeTalkThreshold = ELISTENBEFORETALKTRESHOLD_DEFAULT,
  .iTxPowerLevelMax = APP_MAX_TX_POWER,
  .iTxPowerLevelAdjust = APP_MEASURED_0DBM_TX_POWER,
  .eRegion = APP_FREQ
};
```

**Figure 3: APP_FREQ needs to be set to your RF region**

Refer to Table 1 for a complete list of supported frequencies by the SDK.

***Hint*** Navigate to Silicon Labs website, to see which countries has been approved for the Z-Wave RF.

**Table 1: Overview of a possible frequencies**

| Frequency Region | Variable to use |
|---|---|
| Europe | REGION_EU |
| United States of America | REGION_US |
| Australia/New Zealand | REGION_ANZ |
| Hong Kong | REGION_HK |
| Malaysia | REGION_MY |
| India | REGION_IN |
| Israel | REGION_IL |
| Russia | REGION_RU |
| China | REGION_CN |
| Japan | REGION_JP |
| Korea | REGION_KR |

In this guide we will be using the European frequency, thus we enter "*REGION_EU*".

```
static const SRadioConfig_t RadioConfig =
{
  .iListenBeforeTalkThreshold = ELISTENBEFORETALKTRESHOLD_DEFAULT,
  .iTxPowerLevelMax = APP_MAX_TX_POWER,
  .iTxPowerLevelAdjust = APP_MEASURED_0DBM_TX_POWER,
  .eRegion = REGION_EU
};
```

**Figure 4: APP_FREQ is set to REGION_EU**

### 2.3 Enable Serial Debug

The serial debug interface is useful for printing various information without affecting the flow and timing in the application, such as using a debugger.

1. In the main source file "`SwitchOnOff.c`", locate the variable `DEBUGPRINT` in the include section.
2. Uncomment the define to enable serial debug.

```
/*****************************************************************************/
/*                              INCLUDE FILES                                */
/*****************************************************************************/

#include "config_rf.h"
#include <stdbool.h>
#include <stdint.h>
#include "SizeOf.h"
#include "Assert.h"
#include <SWO_Debug.h>
#include "DebugPrintConfig.h"
#define DEBUGPRINT
#include "DebugPrint.h"
#include "config_app.h"
```

**Figure 5: Make sure to uncomment DEBUGPRINT**

### 2.4 Compile your first Z-Wave Application!

You have now configured the Z-Wave sample application, and you are ready to compile.

1. Click on the "*Build*"  button to start building the project.
2. When the build finishes after a short while, a new folder named "*Binaries*" are showed in the Project Explore. Expand the folder and right click on the *.**hex** file to select "*Flash to Device..*".
3. Select the connected hardware in the pop-up window. The "Flash Programmer" is now prefilled with all needed data, and you are ready to click on "Program". See Figure 6.
4. Click "Program".

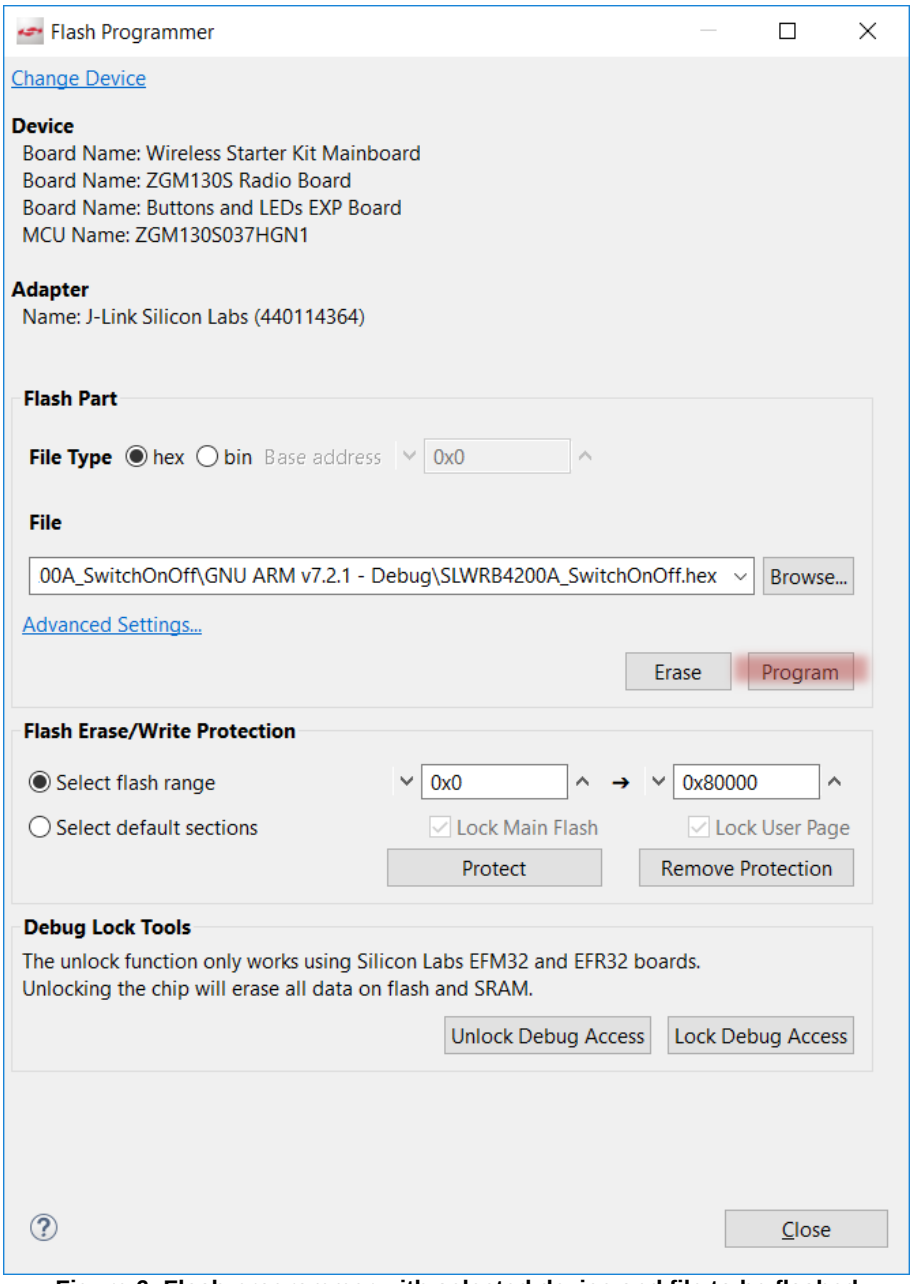


**Figure 6: Flash programmer with selected device and file to be flashed**

After a short while the programming finishes, and your end device is now flashed with a Z-Wave sample application.

# 3 The Serial Debugger

You have now flashed the Z-Wave Sample Application on to your device. In this section we will be including it into a Z-Wave Network and learn how to use the Serial Debugger.

## 3.1 Include into network

In previous exercises we have already included the device into a secure Z-Wave network using SmartStart. Refer to exercise “Include using SmartStart” for instructions.

> ***Hint:*** The internal file system is not erased between reprogramming. This allows a node to stay in a network and keep the same network keys when you reprogram it.
>
> If you need to change (e.g., the frequency at which the module operates or the DSK) you need to “Erase” the chip before the new frequency will be written to the internal NVM.

As such, your device is already included in the network.

## 3.2 Turn the switch On and Off

Test the functionality by verifying you can turn ON and OFF the LED0.

- Test the functionality using the “Basic Set ON” and “Basic Set OFF” in the PC Controller. LED0 should be turning ON and OFF.
- LED0 can also be turned ON and OFF using BTN0 on the hardware.

We have now verified that the device is correct included in the network and that the functionality is still working as expected.

### 3.3 Using the Serial Debugger

The Application Framework features a serial debug connection, enabling you to write statuses, states and values to a terminal.

1. When the device has been flashed, right click on the adapter, click Launch Console.
2. In the Console view, select Serial 1.
3. Click in the input field and press on enter.
4. The small connection icon to the left of the input field should change from to to show you are connected.

The connection is now open, so let's try to see some data.

1. Press the RESET button on the board to see the welcome message.
2. Then press on BTN0 to change the state of the LED0.
3. Use the PC Controller to send a Basic Set ON.

Can you explain the difference in the serial debug log depending on when pressing BTN0 and when clicking on Basic Set ON?

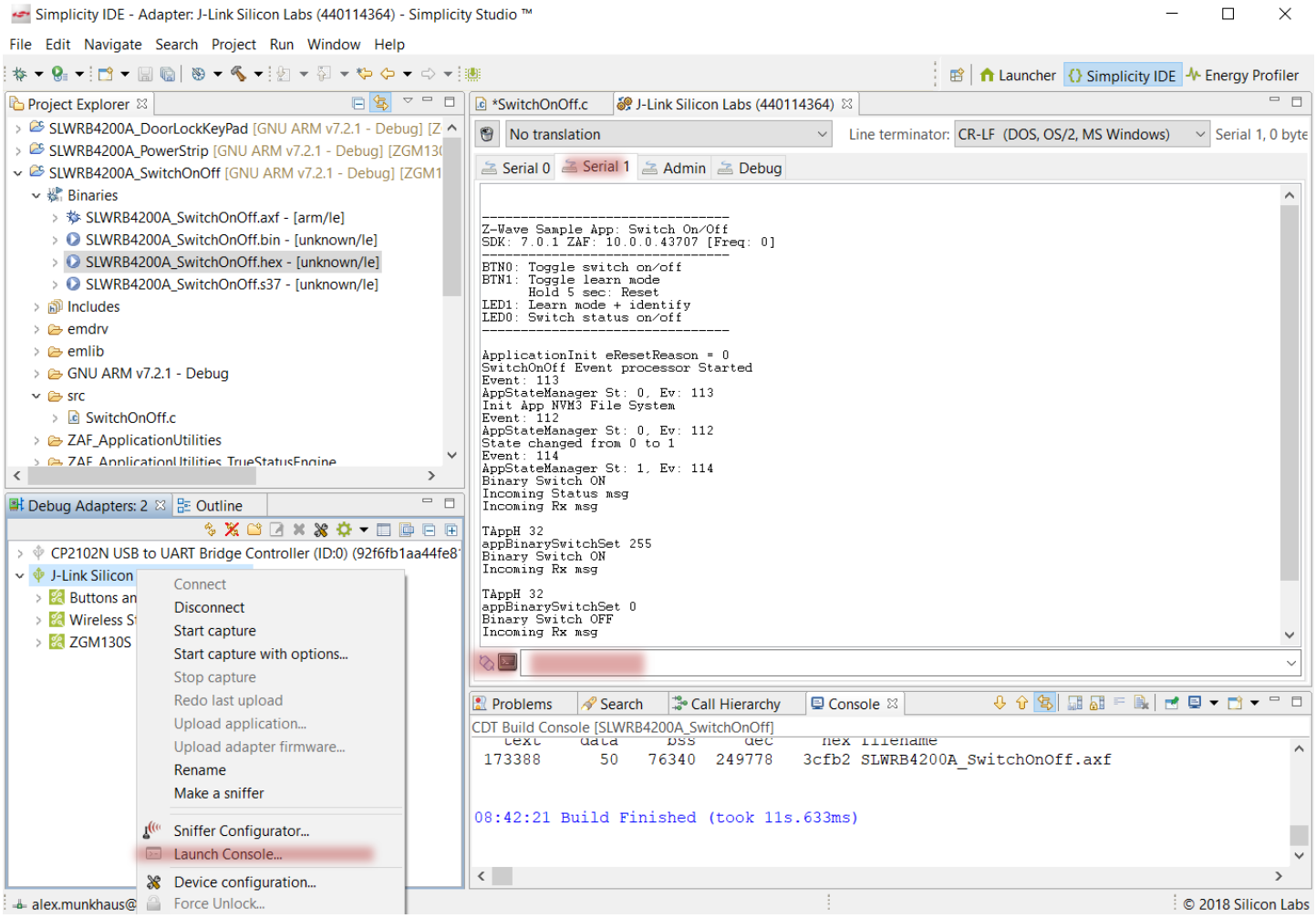


**Figure 7: Serial Debug Interface**

### 3.4 Using Command Class Binary Switch

In previous exercises we covered how command class Basic is mapped to command class Binary Switch, which is being specified by the Z-Wave Device Type specification. Refer to exercise “2 Decrypt Z-Wave RF Frames using the Zniffer” for more details, or the Z-Wave specification SDS14224 Z-Wave Plus v2 Device Type Specification.

**4.5.9.3 Basic Command Class Requirements**

The Basic Command Class MUST be mapped according to Table 17.

**Table 17, Binary Switch Device Type Basic mapping**

| Basic Command | Mapped Command |
|---|---|
| Basic Set (Value) | Binary Switch Set (Value) |
| Basic Report (Current Value, Duration) | Binary Switch Report (Value, Duration). |

**Figure 8: Basic Mapping for Device Type Binary Switch**

In this section, we will try to turn the LED ON and OFF using the Binary Switch command class, and compare the Serial output log when sending Basic command class commands.

1. In the PC Controller, double click on “25 – SWITCH_BINARY” under secure Command Classes in the lower left corner.
2. This opens the “Command Classes” view in the PC Controller and select the Switch Binary Command class.
3. Set the Command to “0x01 – SWITCH_BINARY_SET”
4. Set the “Target Value” to either “00-OFF_DISABLE” or “FF-ON_ENABLE”
5. Click “Send”.

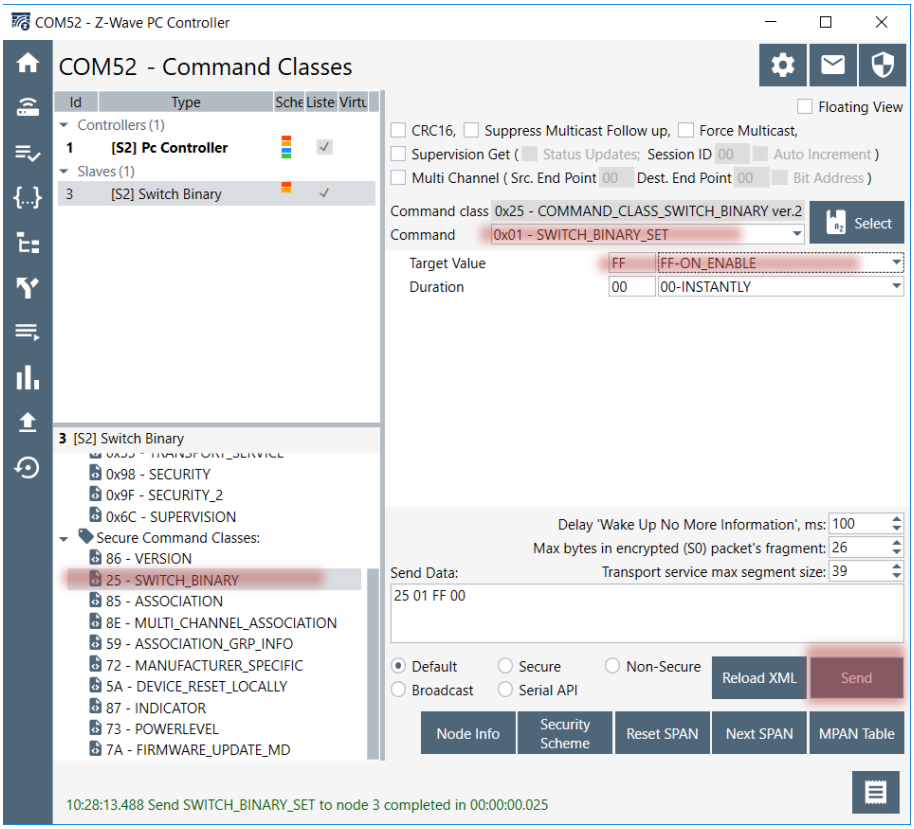


**Figure 9: Command Class View in PC Controller**

Can you explain the difference in the serial debug log depending on when sending a Basic Set and a Binary Switch Set?

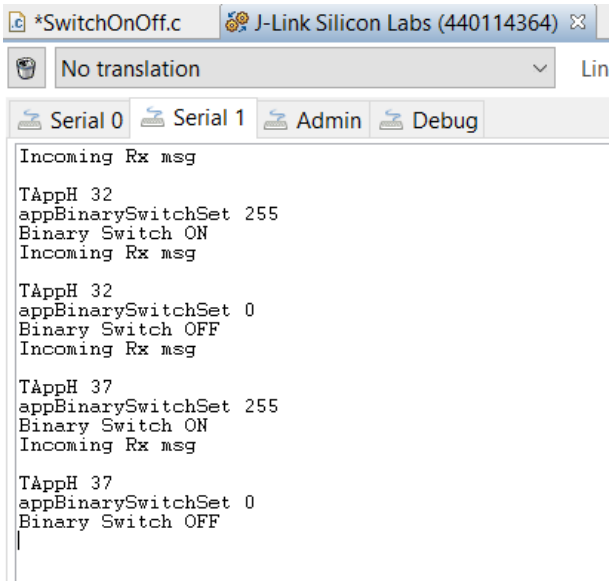


**Figure 10: Serial Debug Log with both Basic Set and Binary Switch Set**

*This concludes the tutorial in how to compile and debug a Z-Wave Sample Application.*