USBFS LPM PSoC4 Code Example

1.0

Features

- USB Link Power Management (LPM) event handling
- Enter/Exit deep sleep mode
- Enter/Exit hibernate mode

General Description

This code example demonstrates the USB Link Power Management (LPM) implementation.

PSoC operates in the active power mode before host decides to suspend the USB device. During enumeration, the device reports two Best Effort Service Latency (BESL) values to the host via/with a binary device object store (BOS) descriptor. Host sends an LPM request to put the device in the suspend mode. The LPM request has a BESL value and a remote wake-up allowance. Based on the BESL value received in the LPM request, the device decides to enter the deep sleep, hibernate mode or stay in the active mode. When the host decides to wake up the USB device, it initiates a resume condition on the bus according to the BESL value sent in the LPM request. When the resume condition is detected PSoC wakes up and changes its power mode to active.

The LED is used to indicate the USB device state and the PSoC power mode.

Development Kit Configuration

This example project runs on the CY8CKIT-046 kit from Cypress Semiconductor. A description of the kit, along with more code examples and ordering information, is at http://www.cypress.com/go/cy8ckit-046.

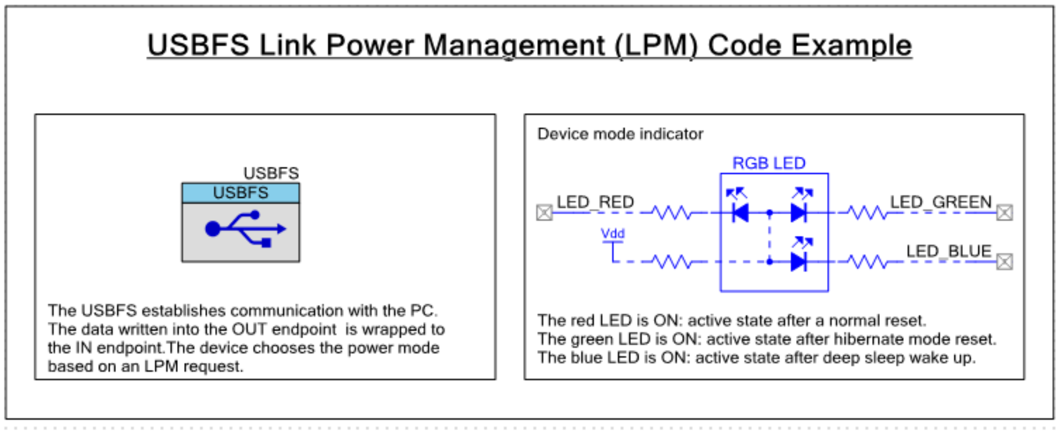
Project Configuration

The example project consists of the USBFS and pins components. The project schematic is in Figure 1.

The USBFS establishes communication with the PC. The USB device has VID = 0x4B4 (this is Cypress Semiconductor vendor ID) and PID = 0x8051 (arbitrarily chosen). In addition, the appropriate string descriptors provide the company's name and the project name. The device is powered through the USB connection, so the device configuration is bus-powered. The device has two BULK endpoints: EP1 IN and EP2 OUT. The OUT endpoint allows the host to write data into the device and the IN endpoint allows the host to read data from the device. Every endpoint maximum packet is 64 bytes, therefore update to 64 bytes can be transferred. BOS and extension descriptors are added to the descriptor tree. These descriptors inform the host that the device supports an LPM request and specify Base BESL and Deep BESL values. Refer to the ECN USB 2.0 Link Power Management Addendum specification for details.

The pin components control the LED which indicates the USB device state and the PSoC power mode. The red LED is on when PSoC is in the active mode. The RGB LED is off when the device is in the deep sleep or hibernate mode. The device turns on the green or blue LED when it returns from the hibernate or deep sleep mode to the active mode accordingly. The red LED is on when PSoC is in the active mode. The RGB LED is off when the device is in the deep sleep or hibernate mode. The device turns the green or blue LED when it returns from the hibernate or deep sleep mode to the active mode accordingly.

Figure 1. Example Project Design Schematic

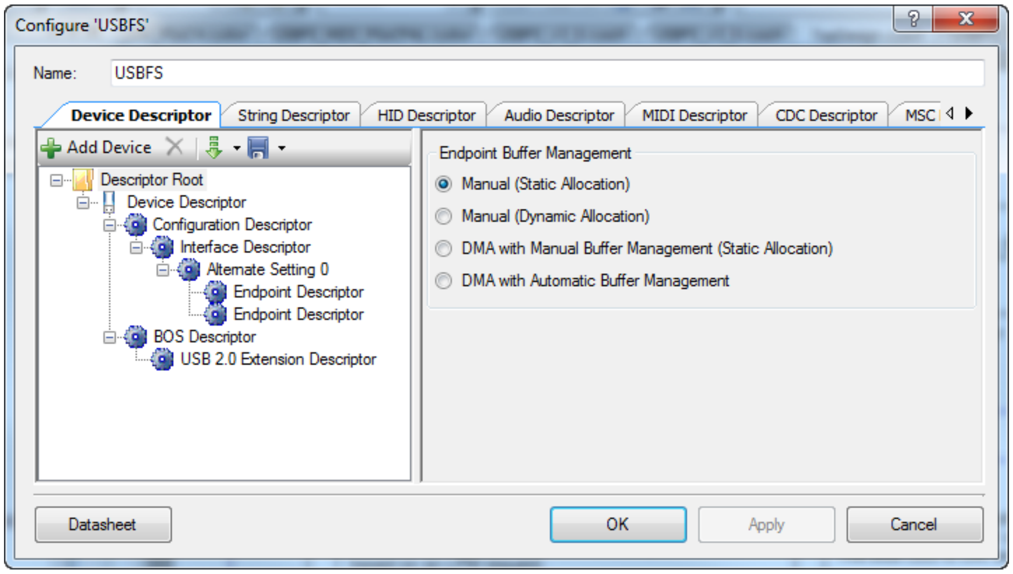


The USBFS establishes communication with the PC. The data written into the OUT endpoint is wrapped to the IN endpoint.The device chooses the power mode based on an LPM request.

The red LED is ON: active state after a normal reset.
The green LED is ON: active state after hibernate mode reset.
The blue LED is ON: active state after deep sleep wake up.

The important USBFS component configuration Tabs are below.

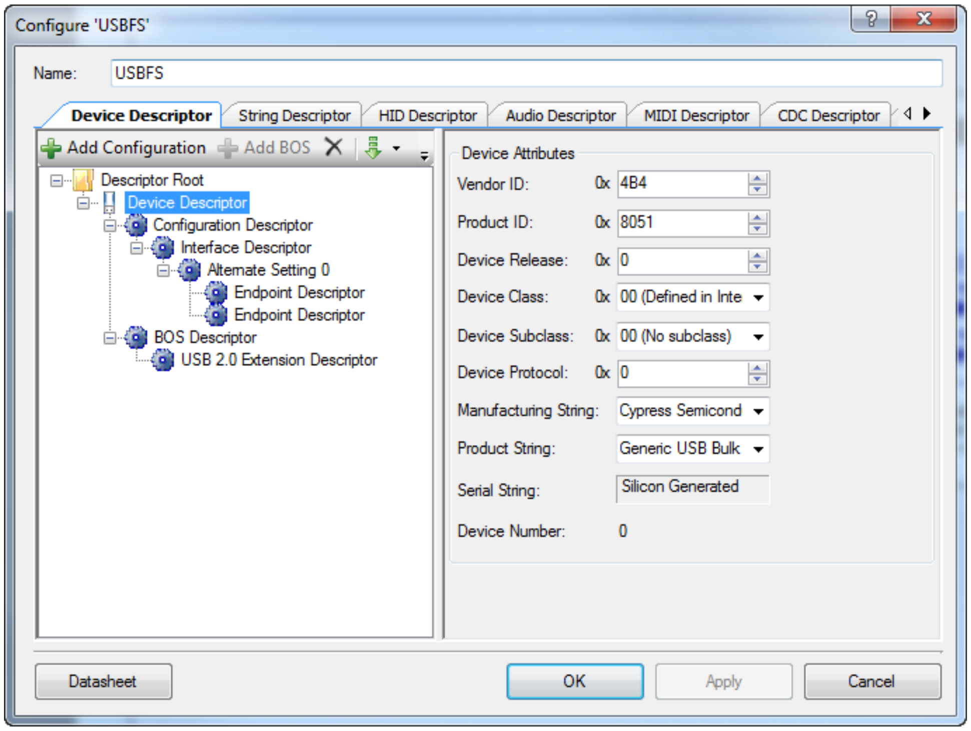
Figure 2. USBFS Descriptor Root



Endpoint Buffer Management

- [✓ Manual (Static Allocation)]
- Manual (Dynamic Allocation)
- DMA with Manual Buffer Management (Static Allocation)
- DMA with Automatic Buffer Management

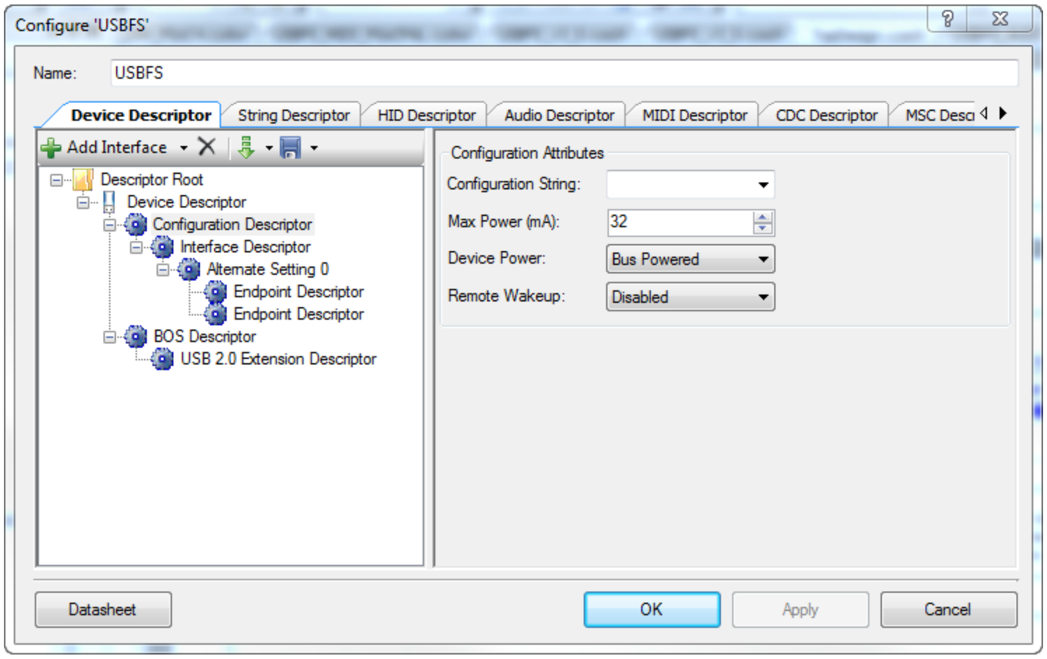
Figure 3. USBFS Device descriptor



Device Attributes

Vendor ID:	0x 4B4
Product ID:	0x 8051
Device Release:	0x 0
Device Class:	0x 00 (Defined in Inte
Device Subclass:	0x 00 (No subclass)
Device Protocol:	0x 0
Manufacturing String:	Cypress Semicond
Product String:	Generic USB Bulk
Serial String:	Silicon Generated
Device Number:	0

Figure 4. USBFS Configuration Descriptor


Configure 'USBFS'

Name: USBFS

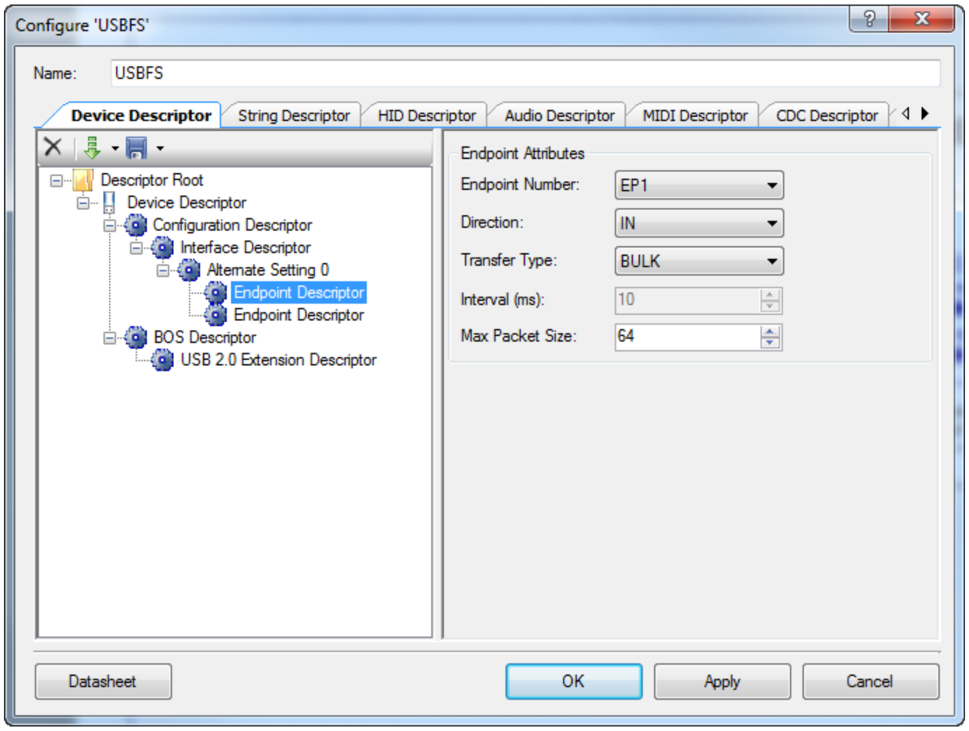
Device Descriptor | String Descriptor | HID Descriptor | Audio Descriptor | MIDI Descriptor | CDC Descriptor | MSC Descr

Descriptor Root
- Device Descriptor
  - Configuration Descriptor
    - Interface Descriptor
      - Alternate Setting 0
        - Endpoint Descriptor
        - Endpoint Descriptor
  - BOS Descriptor
    - USB 2.0 Extension Descriptor

Configuration Attributes

Configuration String:	
Max Power (mA):	32
Device Power:	Bus Powered
Remote Wakeup:	Disabled

Figure 5. USBFS Endpoint 1 Descriptor


Configure 'USBFS'

Name: USBFS

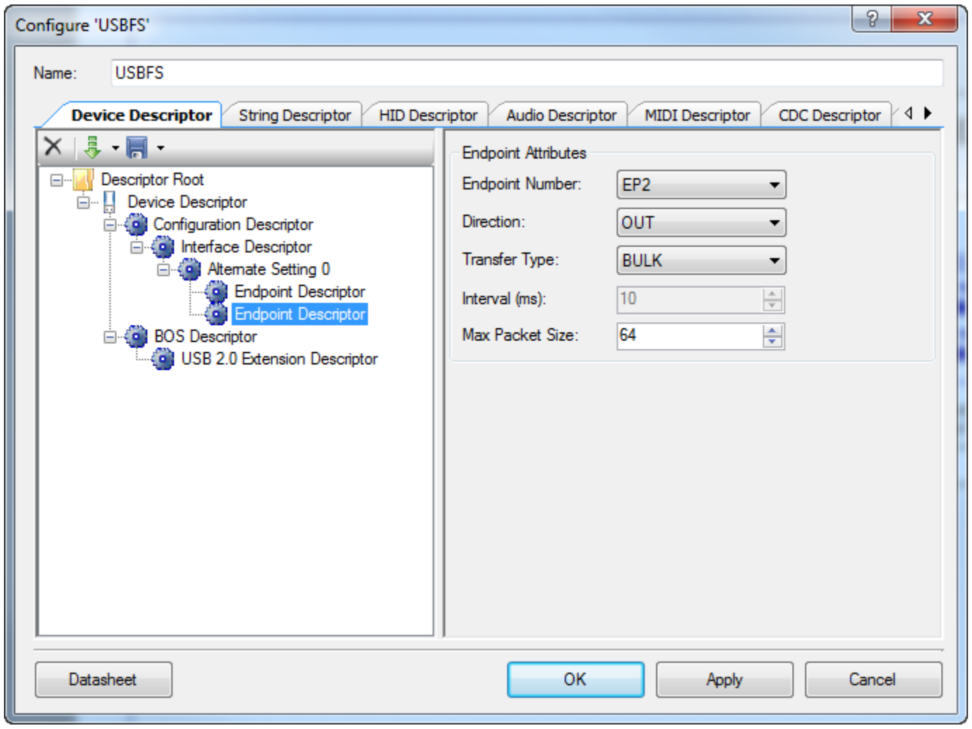
Device Descriptor | String Descriptor | HID Descriptor | Audio Descriptor | MIDI Descriptor | CDC Descriptor

Descriptor Root
- Device Descriptor
  - Configuration Descriptor
    - Interface Descriptor
      - Alternate Setting 0
        - Endpoint Descriptor
        - Endpoint Descriptor
  - BOS Descriptor
    - USB 2.0 Extension Descriptor

Endpoint Attributes

Endpoint Number:	EP1
Direction:	IN
Transfer Type:	BULK
Interval (ms):	10
Max Packet Size:	64

Figure 6. USBFS Endpoint 2 Descriptor



Name: USBFS

Endpoint Attributes

Endpoint Number: EP2

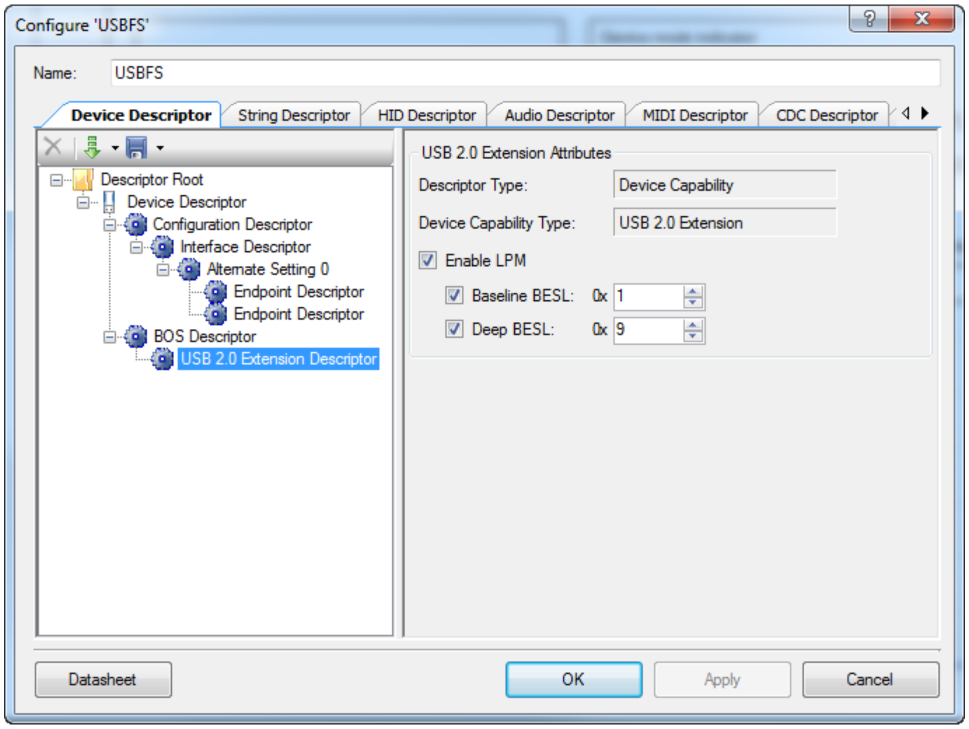
Direction: OUT

Transfer Type: BULK

Interval (ms): 10

Max Packet Size: 64

Figure 7. USBFS Extension Descriptor



Name: USBFS

USB 2.0 Extension Attributes

Descriptor Type: Device Capability

Device Capability Type: USB 2.0 Extension

[✓] Enable LPM

[✓] Baseline BESL: 0x 1

[✓] Deep BESL: 0x 9

Project Description

The USBFS component configures for operation and starts in the main firmware routine. The code waits for the device enumeration. After enumeration, the device runs in the active mode and switches between two tasks.

- Task 1. (Bulk wraparound) The OUT endpoint buffer is checked for the data availability and the received data is loaded into the IN endpoint buffer (if there is space). If the IN endpoint buffer is full, the data remains in the RAM buffer until the IN endpoint buffer becomes empty.
- Task 2. (Detection of LPM requests) The USB device decides to deep sleep/hibernate or remain in the active mode depending on the BESL value from the LPM request. If the BESL value is smaller than the Base BESL, the device remains in the active mode. If the BESL value is greater or equal to the Base BESL and smaller than Deep, the BESL device goes to the deep sleep mode. If the BESL is greater or equal to the Deep BESL, the device goes to the hibernate mode.

Deep Sleep

The USBFS component stores its current configuration and sets up a wake-up source calling the USBFS_Suspend() function before going to the low-power mode. The wake-up source is the falling edge of the USB D+ pin.

When the host wants to wake the device up after suspend, it does so by reversing the polarity of the signal on the data lines for a time at least equal to the BESL value from the LPM request. The “low-speed end of the packet” completes the signal. After the D+ pin polarity is reverted, PSoC wakes up because a wake-up event occurs (the falling edge on the D+ pin). Call of the USBFS_Resume() API restores the USBFS component configuration and the USB device is ready for the operation. The device returns into the active mode and executes its tasks after completion of all restoring APIs for other components.

Note The USBFS data endpoints require re-initialization after wakeup from the low-power mode, which includes enabling the OUT endpoints and loading the IN endpoints. The data available in the data IN or OUT endpoints does not retain in the low-power mode and is lost. It is the application responsibility to manage this properly.

Hibernate

The hibernate mode requires additional steps to backup values of internal data structures and registers. After waking from the hibernate mode, the device resets and starts executing firmware from the beginning. The HibernateBackUp() function stores the device address, configuration and the EP date structures in the NO_INIT section to restore it after the hibernate. The API CySysPmGetResetReason() distinguishes a start after a normal reset and a start from the hibernate mode reset (for details see Help->System Reference Guides for PSoC4). After wakeup, the registers and internal data structures needs restoring. The HibernateRestore(void) function calls USBFS_Init() and USBFS_InitComponent() forms the UBSFS_Start() function to initialize all the registers and restore internal data structures. This function also calls the

USBFS_ConfigReg() function to restore the registers according to the configuration which was before the hibernate. We also need to restore the device address as the last step, because the USBFS_InitComponent() function resets the device address to 0.

Note During restoring after the hibernate until calling the USBFS_InitComponent() function, the device does not drive the D+ signal high. This period consists of the wake-up time from the hibernate mode for the PSoC 4200L device (refer to device datasheet: Help->Documentation-> Device Datasheets: PSoC 4200L datasheet) and resume-code execution time. During this period, the host should keep the resume condition on the bus. According to the LPM specification, the host drives a resume condition according to the BESL value in the LPM request to the device (see table X-X1 of Errata for USB 2.0 ECN: Link Power Management (LPM) - 7/2007 for correspondence between time and BESL value). If the time of the resume condition on the bus is shorter than the device wake-up time, the host detects a reset condition.

LPM Work Flow

The basic workflow algorithm of the LPM:**(Figure 8).**

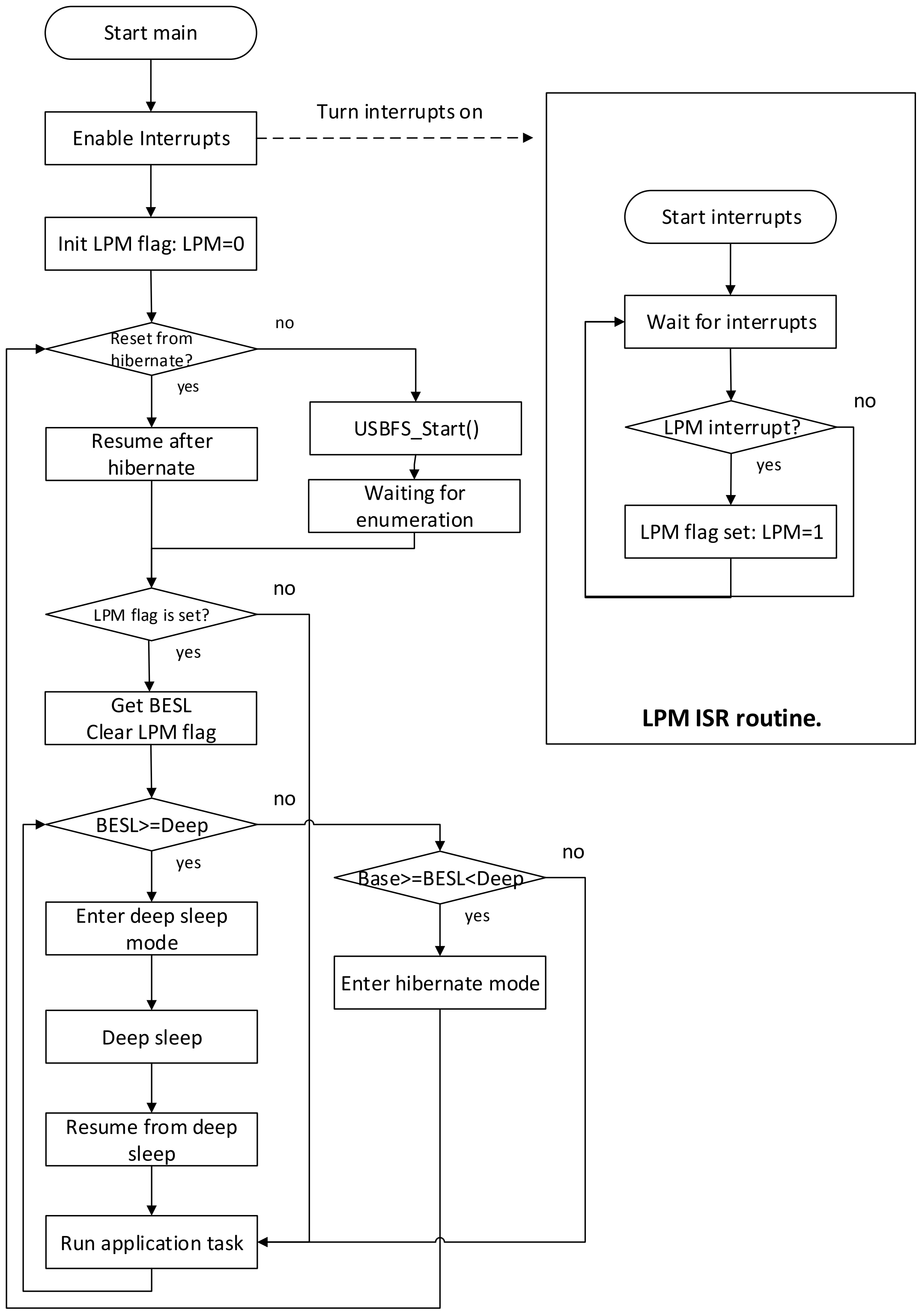
- Setup routine:
  - An LPM request specifies the max wake-up time in the BESL value. Setting Base BESL and Deep BESL values for your design in the extension descriptor informs the host about for how long the device resumes from deep sleep and hibernate.
  - Create the Interrupt Service Routine (ISR) callback function. The LPM ISR calls this function after sending an ACK to the LPM request.
    Note We can set/get a current response to an LPM request (ACK, NYET or NACK) by using the API: USBFS_Lpm_SetResponse()/USBFS_Lpm_GetResponse() (see USBFS datasheet).
  - Update a global flag that we have a new LPM request in the LPM ISR callback function.
    Note We recommend keeping the ISR as short as possible and do not process the LPM request inside the ISR callback.
- The main routine performs the main application action and periodically checks the LPM flag. Call the Suspend routine if the LPM flag is set.
- LPM routine:
  - Read the BESL value from the received LPM request using the USBFS_Lpm_GetBeslValue() API.
  - Choose the power mode basing on the received BESL value. If the LPM BESL is greater or equal to the Deep BESL (assuming that the Deep BESL is greater than the Base BESL), then we have enough time to put the device in the hibernate mode. If the LPM BESL is greater than the Base BESL, but is smaller than the Deep BESL, we can put the device into the deep dleep mode. If the LPM BESL is

smaller than the Base BESL, we should stay in the active mode, because we don't not have enough time to wake up.

- Put the device into the selected power mode.

- Resume routine:
    - From deep sleep. The standard procedure of resume is to call the resume APIs of all suspended components.
    - From hibernate. The device restarts and starts from the beginning of the main. You need to reinitialize the components. An example of the USBFS initialization is in the HibernateBackUp()/HibernateRestore() functions of the main.c file.
    - Remote wake up. Check if the host allows a remote wake-up in the LPM request with the USBFS_Lpm_RemoteWakeUpAllowed() API and perform a remote wake-up if it returns true.

Figure 8. LPM Basic Work Flow **Workflow**



Example Project Execution Flow

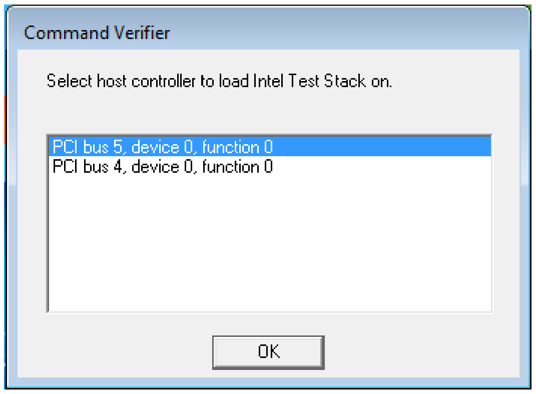
To execute the USBFS component code example you need the following equipment:

- The USB 3.0 hub with the LPM support (for example Fresco USB 3.0 xHCI-based PDK for SupersSpeed USB recommended on usb.org)
- The USB3CV tool Ver 2.0.2.2 installed (You can download it from USBCV.ORG site)

Follow the procedure below:

1. Connect the PSoC kit to the PC through a USB connector for programming.
2. Build the USBFS_LPM_PSoC4 example project and program into the device. At this point, you can disconnect the USB cable from the programming connector.
3. Run the USB3CV tool. Select the USB3.0 hub with the LPM support (**Figure 9**).
4. Select Chapter 9 Tests [USB 2 devices]. Choose the debug mode. Select TD 9.21: LPM L1 Suspend Resume Test (**Figure 10**).
5. Connect the kit to the PC's hub selected in item 3 through a USB connector for communication.
6. Click the **Run** button and select the device with VID=04B4 and PID =8051 (**Figure 11**).
7. Choose the configuration index 0x0 (**Figure 12**). It is number of the device configuration descriptor in our descriptors list, which supports LPM.

Figure 9. USB3CV Tool Host Controller Selection

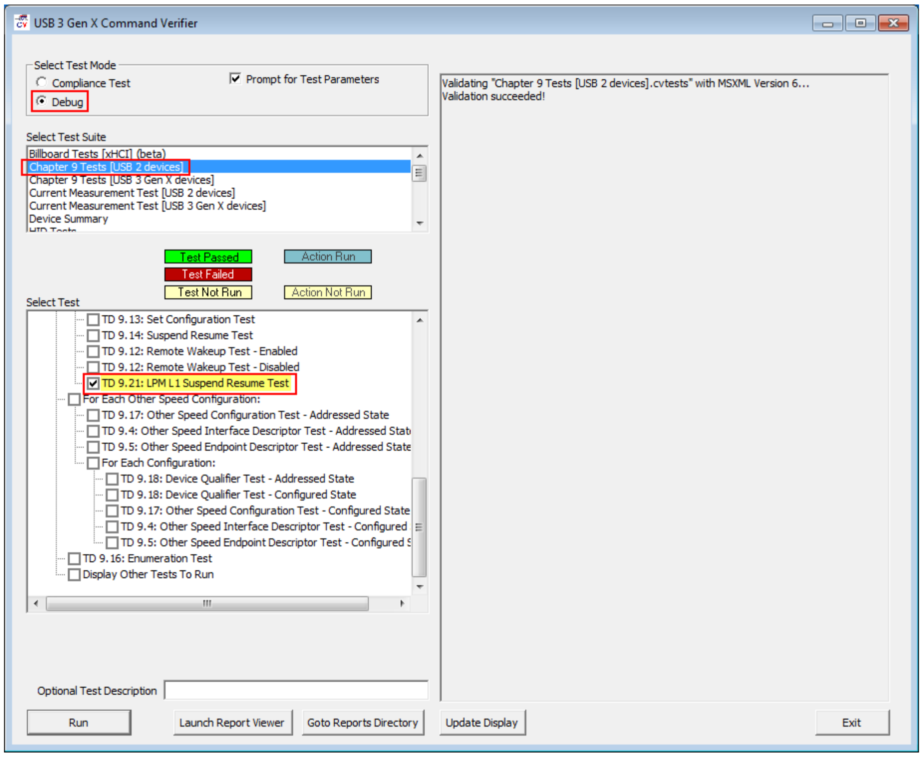


Command Verifier

Select host controller to load Intel Test Stack on.

PCI bus 5, device 0, function 0
PCI bus 4, device 0, function 0

Figure 10. USB3CV LPM Test


USB 3 Gen X Command Verifier

Select Test Mode
- Compliance Test
- [✓ Debug]

☑ Prompt for Test Parameters

Select Test Suite
- Billboard Tests [xHCI] (beta)
- [✓ Chapter 9 Tests [USB 2 devices]]
- Chapter 9 Tests [USB 3 Gen X devices]
- Current Measurement Test [USB 2 devices]
- Current Measurement Test [USB 3 Gen X devices]
- Device Summary
- HID Tests

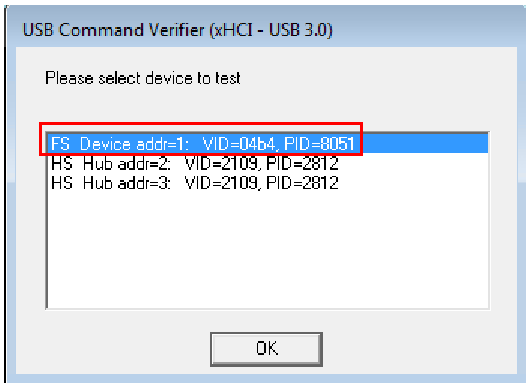
🟩 Test Passed
🟥 Test Failed
🟨 Test Not Run
🟦 Action Run
🟨 Action Not Run

Select Test
- ☐ TD 9.13: Set Configuration Test
- ☐ TD 9.14: Suspend Resume Test
- ☐ TD 9.12: Remote Wakeup Test - Enabled
- ☐ TD 9.12: Remote Wakeup Test - Disabled
- ☑ TD 9.21: LPM L1 Suspend Resume Test
- ☐ For Each Other Speed Configuration:
  - ☐ TD 9.17: Other Speed Configuration Test - Addressed State
  - ☐ TD 9.4: Other Speed Interface Descriptor Test - Addressed State
  - ☐ TD 9.5: Other Speed Endpoint Descriptor Test - Addressed State
  - ☐ For Each Configuration:
    - ☐ TD 9.18: Device Qualifier Test - Addressed State
    - ☐ TD 9.18: Device Qualifier Test - Configured State
    - ☐ TD 9.17: Other Speed Configuration Test - Configured State
    - ☐ TD 9.4: Other Speed Interface Descriptor Test - Configured
    - ☐ TD 9.5: Other Speed Endpoint Descriptor Test - Configured S
- ☐ TD 9.16: Enumeration Test
- ☐ Display Other Tests To Run

```
Validating "Chapter 9 Tests [USB 2 devices].cvtests" with MSXML Version 6...
Validation succeeded!
```

Optional Test Description

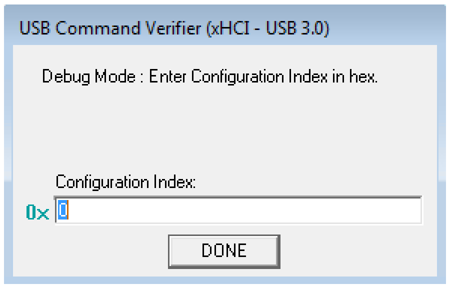
Figure 11. USB3CV Device Selection


USB Command Verifier (xHCI - USB 3.0)

Please select device to test

```
FS  Device addr=1:   VID=04b4, PID=8051
HS  Hub addr=2:   VID=2109, PID=2812
HS  Hub addr=3:   VID=2109, PID=2812
```

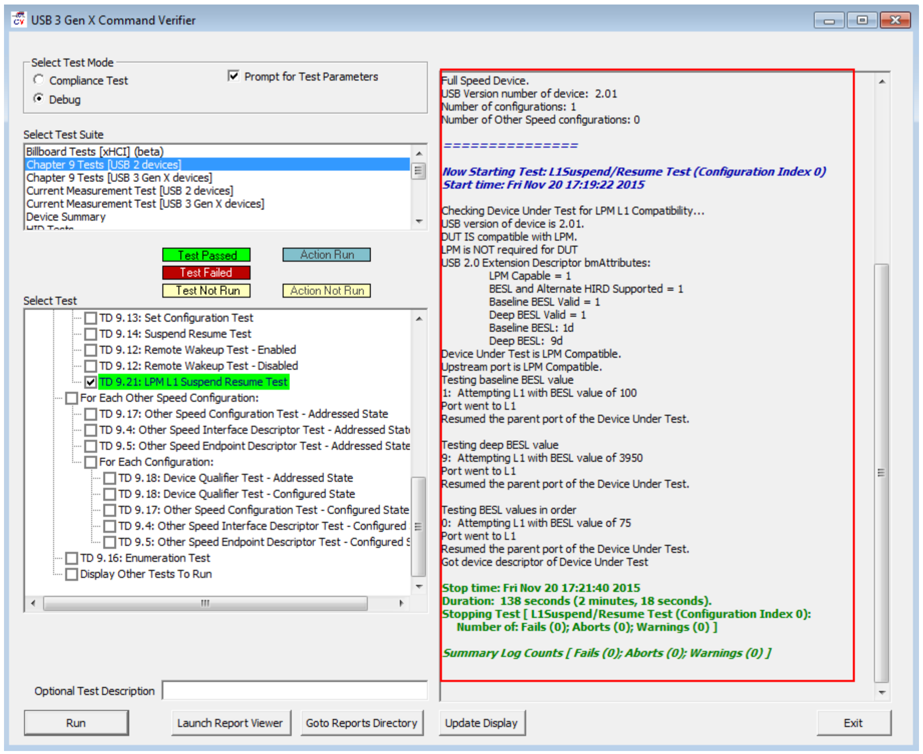
Figure 12. USB3CV Device For Test Selection


USB Command Verifier (xHCI - USB 3.0)

Debug Mode : Enter Configuration Index in hex.

Configuration Index: 0x 0

Figure 13. USB3CV Results



Select Test Mode
( ) Compliance Test
(●) Debug
☑ Prompt for Test Parameters

Select Test Suite
- Billboard Tests [xHCI] (beta)
- Chapter 9 Tests [USB 2 devices]
- Chapter 9 Tests [USB 3 Gen X devices]
- Current Measurement Test [USB 2 devices]
- Current Measurement Test [USB 3 Gen X devices]
- Device Summary

Select Test
- ☐ TD 9.13: Set Configuration Test
- ☐ TD 9.14: Suspend Resume Test
- ☐ TD 9.12: Remote Wakeup Test - Enabled
- ☐ TD 9.12: Remote Wakeup Test - Disabled
- ☑ TD 9.21: LPM L1 Suspend Resume Test
- ☐ For Each Other Speed Configuration:
  - ☐ TD 9.17: Other Speed Configuration Test - Addressed State
  - ☐ TD 9.4: Other Speed Interface Descriptor Test - Addressed Stat
  - ☐ TD 9.5: Other Speed Endpoint Descriptor Test - Addressed State
  - ☐ For Each Configuration:
    - ☐ TD 9.18: Device Qualifier Test - Addressed State
    - ☐ TD 9.18: Device Qualifier Test - Configured State
    - ☐ TD 9.17: Other Speed Configuration Test - Configured State
    - ☐ TD 9.4: Other Speed Interface Descriptor Test - Configured
    - ☐ TD 9.5: Other Speed Endpoint Descriptor Test - Configured
- ☐ TD 9.16: Enumeration Test
- ☐ Display Other Tests To Run

```
Full Speed Device.
USB Version number of device:  2.01
Number of configurations: 1
Number of Other Speed configurations: 0

===============

Now Starting Test: L1Suspend/Resume Test (Configuration Index 0)
Start time: Fri Nov 20 17:19:22 2015

Checking Device Under Test for LPM L1 Compatibility...
USB version of device is 2.01.
DUT IS compatible with LPM.
LPM is NOT required for DUT
USB 2.0 Extension Descriptor bmAttributes:
            LPM Capable = 1
            BESL and Alternate HIRD Supported = 1
            Baseline BESL Valid = 1
            Deep BESL Valid = 1
            Baseline BESL: 1d
            Deep BESL:  9d
Device Under Test is LPM Compatible.
Upstream port is LPM Compatible.
Testing baseline BESL value
1:  Attempting L1 with BESL value of 100
Port went to L1
Resumed the parent port of the Device Under Test.

Testing deep BESL value
9:  Attempting L1 with BESL value of 3950
Port went to L1
Resumed the parent port of the Device Under Test.

Testing BESL values in order
0:  Attempting L1 with BESL value of 75
Port went to L1
Resumed the parent port of the Device Under Test.
Got device descriptor of Device Under Test

Stop time: Fri Nov 20 17:21:40 2015
Duration:  138 seconds (2 minutes, 18 seconds).
Stopping Test [ L1Suspend/Resume Test (Configuration Index 0):
   Number of: Fails (0); Aborts (0); Warnings (0) ]

Summary Log Counts [ Fails (0); Aborts (0); Warnings (0) ]
```

Expected Results

You observe that the RGB LED is turning ON/OFF during the test. At the end of the USB3CV test, the color should be green. The green LED indicates that the device returns to the active mode from the hibernate mode (see Table 1). The CV test should PASS, which indicates that the device returns a descriptor after resuming from deep sleep and hibernate modes (**Figure 14**).

Table 1. LPM code example LED colors description

The LED color	**Description**
All LED’s off	The device is in the hibernate or deep sleep mode.
The red LED is on	The device is in the active mode. The device enters the active mode from a normal reset.
The green LED is on	The device is in the active mode. The device enters the active mode from a reset after the hibernate mode.
The blue LED is on	The device is in the active mode. The device enters the active mode from deep sleep.

Note Some hubs do not send Deep BESL requests and the device turns ON the blue LED because the device returns only from the deep sleep mode.

Note Some hubs do not support the updated LPM spec and drive a resume request maximum for 1ms. This duration is too short to resume from the hibernate mode for the PSoC 4200L device. The device fails the CV LPM test with such a hub.

Figure 14. PSoC 4200L Pioneer Kit