Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

CE223305 – PSoC 6 MCU: USB Suspend and Resume

Objective

This example demonstrates how PSoC® 6 MCU detects a USB suspend condition, enters a low power state, and restores normal operation when USB activity resumes.

Requirements

Tool: PSoC Creator™ 4.2, Peripheral Driver Library (PDL) 3.1.0

Programming Language: C

Associated Parts: All PSoC 6 MCU parts with USB

Related Hardware: PSoC 6 WiFi-BT Pioneer Kit

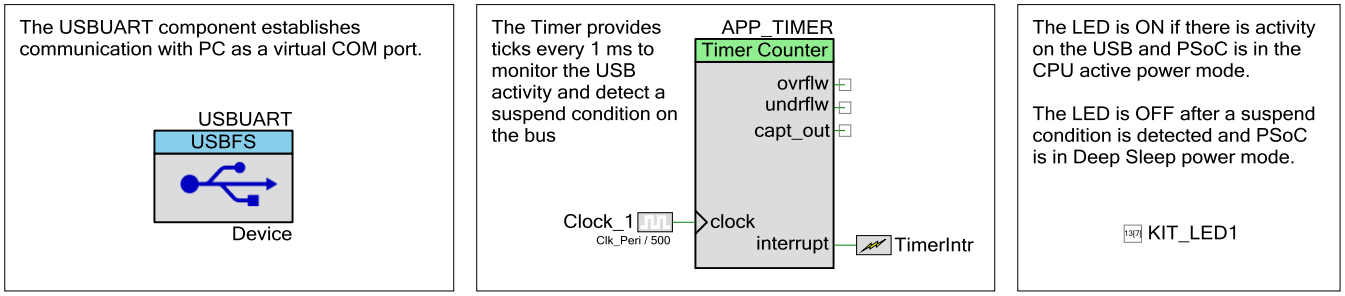
Overview

PSoC 6 MCU operates in the CPU active power mode before the host decides to suspend the USB device. The PSoC 6 MCU goes to Deep Sleep as soon as a suspend condition is detected, to consume less power. The USB device maximum suspend current should not exceed 500 µA (for a low-power device) to be compliant with the USB specification. When the host decides to wake up the USB device, it drives a resume condition on the bus. This drives the PSoC 6 MCU device to wake up and change its power mode to CPU active.

The example uses the kit LED to indicate the USB device state as well as the PSoC 6 MCU power mode. The LED is OFF when a suspend condition is detected and PSoC 6 MCU is in Deep Sleep. The LED is ON when the USB is active and PSoC 6 MCU is in CPU active power mode.

Figure 1 shows the PSoC Creator schematic for this project.

Figure 1. Project Schematics



Hardware Setup

This example uses the kit's default configuration. Refer to the kit guide to ensure that the kit is configured correctly.

Software Setup

If necessary, install a terminal emulator like Tera Term or PuTTY on your computer. The example uses the terminal window to send and display messages.

Operation

1. Connect the kit board to your PC using the provided USB cable through the KitProg USB connector.

2. Build the project and program it into the PSoC 6 MCU device. Choose **Debug** > **Program**. For more information on device programming, see PSoC Creator Help. Flash for both CPUs is programmed in a single program operation.
3. Connect another USB cable (or reuse the same cable used to program the kit) to the USB device connector.
4. In the PC, verify that a new USB device enumerates as a Virtual COM port.

   Note: Windows 7 requires that you install a USB driver for this code example. Open the Device Manager, find this USB device in the **Other devices** branch of the tree. Open the context menu and select **Update Driver Software**. Browse to the INF file from the project root as the driver to be installed.
5. Run any terminal emulator and make a new connection to the Virtual COM port. Observe the message “USB is active” periodically printed in the terminal. Observe that the kit LED is ON.
6. To put the device into suspend mode, on a Windows PC go to the Start menu and put the PC into sleep mode. This makes the USB traffic stop; PSoC 6 MCU enters the suspend mode, and the LED turns OFF.
7. Use the mouse or keyboard to wake up the PC. The host generates a resume condition on the USB; PSoC 6 MCU resumes and restores communication through USB (the device is not enumerated again). Observe that the LED in the kit is ON again.
8. After the PC wakes up, the communication with the USBFS device can be observed as described in Step 5.

Design and Implementation

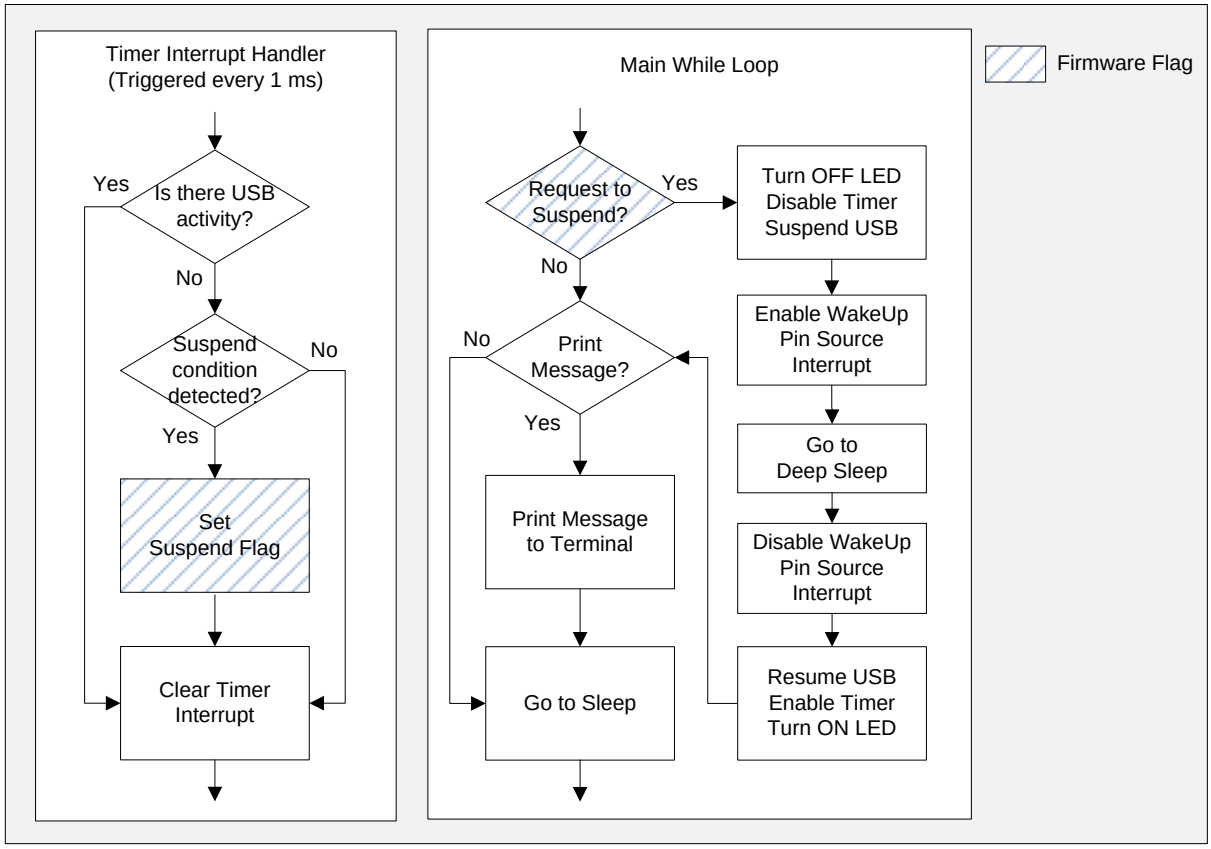
In the main firmware routine, the USBFS block is configured to use the Communication Device Class (CDC). After enumeration, the device periodically checks for activity every 1 ms. The USB device is suspended if there is no activity on the bus for longer than 3 ms. Normally, Start-Of-Frame (SOF) packets are sent by the host every 1 ms. The USBFS driver layer is not capable of tracking the suspend condition by itself. Therefore, a timer is placed in the design to provide ticks each 1 ms. The USB activity is checked every tick to identify when bus is idle for 3 ms.

After the USB suspend condition is detected, PSoC 6 MCU leaves the CPU active power mode and goes to Deep Sleep. This allows the device to reduce power consumption while the host does not communicate with it. Before going to Deep Sleep, the LED is turned OFF, the timer is disabled, and the wakeup source interrupt is enabled. The wakeup source is the falling edge on the USB Dp pin. The `Cy_USBFS_Dev_Drv_Suspend()` function is also called to prepare the USB hardware block to enter suspend mode.

When the host wants to wake the device up after the suspend condition, it does so by reversing the polarity of the signal on the data lines for at least 20 ms. The signal is completed with the low-speed end of the packet signal. As soon as the Dp pin polarity is reverted, PSoC 6 MCU wakes up because a wakeup event occurs (falling edge on the Dp pin). The USBFS hardware block configuration is restored by calling the `Cy_USBFS_Dev_Drv_Resume()` function. The timer is re-enabled, the LED is turned ON, and the wakeup source interrupt is disabled.

Figure 2 shows the firmware flowchart of this code example.

Figure 2. Firmware Flowchart



In CPU active power mode, the device periodically prints a message in the terminal to show that the device firmware and USB connectivity is working. When idle, the device goes to sleep and wakes up at the next USB interrupt.

Components and Settings

Table 1 lists the PSoC Creator Components used in this example, how they are used in the design, and the non-default settings required so they function as intended.

Table 1. PSoC Creator Components

Component	Instance Name	Purpose	Non-default Settings
USBFS	USBUART	Implements the CDC Device Class	All default
TCPWM	APP_TIMER	Tracks the activity on the USB	Period: 100 Interrupt Source: Overflow/Underflow
Clock	Clock_1	Clock sourced to the timer	Divider: 500
Interrupt	TimerIntr	Timer Interrupt	All default
Digital Pin Output	KIT_LED1	Controls the kit LED	All default

To visualize the USBFS descriptor, right-click the USBFS Component and select **Launch USB Configurator**. You can also refer to the *USBUART_cfg.h* file in the *Generated_Source* folder.

To achieve the 0.25% accuracy required by the USB, the IMO needs to be trimmed with USB, as shown in Figure 3. The IMO is configured in the Design Wide Resources clock tab.

Figure 3. Source Clocks Configuration

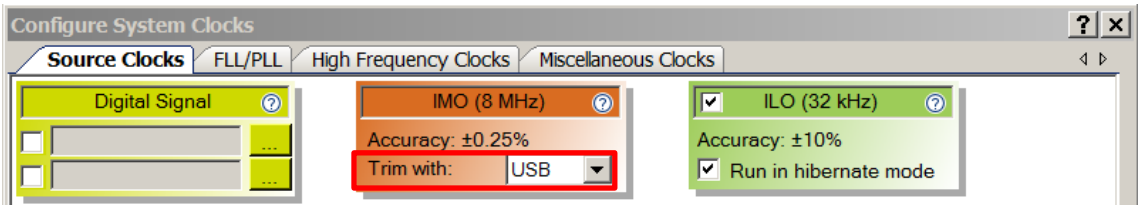


Figure 4 shows PLL/FLL configuration in the Design Wide Resources clock tab. Note that FLL and PLL are used in this application. FLL is used to clock the CPUs. PLL is used to clock the USBFS block.

Figure 4. FLL/PLL Configuration

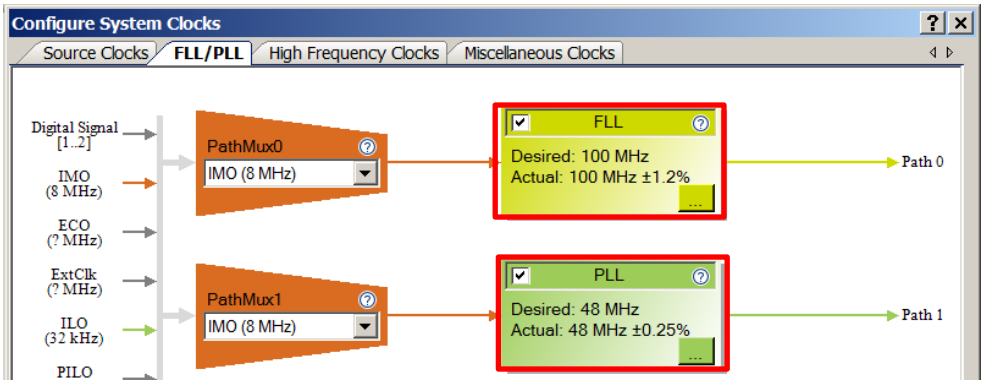
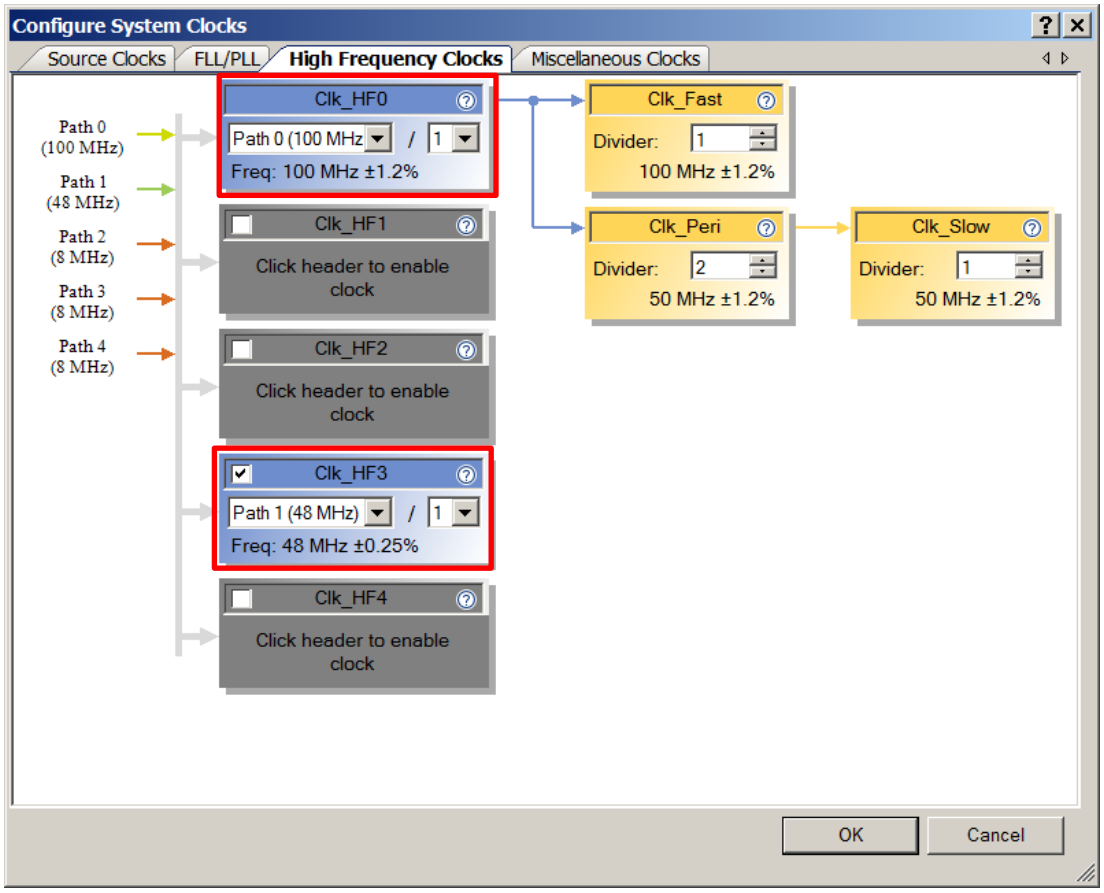


Figure 5 shows the high-frequency clock settings.

Figure 5. High Frequency Clock Configuration



For information on the hardware resources used by a Component, see the Component datasheet.

Reusing This Example

This example is designed for the CY8CKIT-062-WiFi-BT pioneer kit. To port the design to a different PSoC 6 MCU device and/or kit, change the target device using the Device Selector and update the pin assignments in the Design Wide Resources Pins settings as needed.

In some cases, a resource used by a code example (for example, USBFS block) is not supported on another device. In that case, the example will not work. If you build the code targeted at such a device, you will get errors. See the device datasheet for information on what a particular device supports.

Related Documents

Application Notes	
AN221774 Getting Started with PSoC 6 MCU	Describes the PSoC 6 MCU devices and how to build your first PSoC project.
AN215656 PSoC 6 MCU Dual-CPU System Design	Describes the dual-CPU architecture in PSoC 6 MCU, and shows how to build a simple dual-CPU design.
AN219528 PSoC 6 MCU Low-Power Modes and Power Reduction Techniques	Describes the PSoC 6 power modes to optimize power consumption.
Code Examples	
Visit the Cypress Code Example site for a comprehensive collection of code examples using PSoC Creator IDE.	
Device Documentation	
PSoC 6 MCU Datasheets	PSoC 6 MCU Technical Reference Manuals
Development Kit Documentation	
CY8CKIT-062-WiFi-BT PSoC 6 WiFi-BT Pioneer Kit	

Document History

Document Title: CE223305 – PSoC 6 MCU: USB Suspend and Resume

Document Number: 002-23305

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6437874	RLOS	05/07/2019	New code example

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

Arm® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6 MCU

Cypress Developer Community

Community | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709