CE226306 – PSoC 6 MCU Power Measurements

Objective

This example shows how to achieve the power measurements listed in the PSoC® 6 MCU datasheets.

Requirements

Tool: PSoC Creator™ 4.2; Peripheral Driver Library (PDL) 3.1

Programming Language: C (Arm® GCC 5.4.1 and Arm MDK 5.22)

Associated Parts: All dual-CPU PSoC 6 MCU parts

Related Hardware: CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit and CY8CKIT-062-WiFi-BT PSoC 6 WiFi-BT Pioneer Kit

Overview

This example shows how to configure PSoC 6 MCU devices to run the clock frequencies and system modes specified in the Device Level Specifications table in PSoC 6 MCU datasheets. After building and programming the application into PSoC 6 MCU devices, you can measure the current the PSoC 6 MCU core consumes and compare it against the values specified in the datasheet. You can select the configuration by changing a `#define` in firmware.

Hardware Setup

This example requires an ammeter to measure the current consumed by the CM0+ and CM4 CPU cores of the PSoC 6 MCU. When using one of the PSoC 6 Pioneer kits, simply remove the jumper PWR_MON from the PSoC 6 kit and connect the ammeter on the header pins. This jumper is on the bottom side of the kit. Refer to the kit guide for the exact location. If using any other PSoC 6 kit, refer to the kit's user guide for information on how to measure the device's current.

Software Setup

None.

Operation

1. Remove the PWR_MON jumper on CY8CKIT-062 and connect an ammeter to the header pins.
2. Plug the CY8CKIT-062 kit board into your computer's USB port.
3. In the project, open the *specs.h* file. Edit the `#define SPEC_ID` to be one of the specification IDs listed in the datasheet. The same header file also lists all the specification IDs supported. You might also edit the `#define VCCD_SUPPLY` to `BUCK` or `LDO`.
4. 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 memory for both CPUs is programmed in a single program operation.
5. You might also change the PSoC 6 MCU voltage supply. CY8CKIT-062 kits allow choosing 3.3 V or 1.8 V. You should unplug the power to the kit before changing the voltage supply.
6. Observe the current measured by the ammeter. Compare it against the values specified in the datasheet and confirm that the current is below the maximum current. Note that some of the components from the kit attached to the PSoC 6 can cause some current leakage, increasing the overall current consumption and exceeding the maximum specification. This is more visible when the PSoC is in System Deep Sleep and Hibernate modes.

Design and Implementation

PSoC 6 MCU datasheets contain the Device Level-Specification tables, which list several possible configurations of CPU clock frequencies, system power modes, cache support, VCCD supply type, and device voltage supply. Each configuration is linked to a specification ID. The firmware stores details of each specification ID, so you can simply choose the desired configuration by setting the `#define SPEC_ID` macro in the *specs.h* file. Each configuration groups a list of `#defines` as described in Table 1:

Table 1. Configuration Options

Configuration	Options	Description

System Mode	SYSTEM_LP
	SYSTEM_ULP
		Defines the system mode the firmware enters – System Low Power or System Ultra-Low Power modes. The following functions are used:
		`Cy_SysPm_SystemEnterLp(); // Enter System Low Power mode`
		`Cy_SysPm_SystemEnterUlp(); // Enter System Ultra-Low Power mode`

Core Voltage Supply	VCCD_1V1
	VCCD_0V9
		Defines the core voltage supply for the BUCK or LDO – 0.9 V or 1.1 V.
		`Cy_SysPm_SwitchToSimoBuck();`
		`Cy_SysPm_SimoBuckSetVoltage1(CY_SYSPM_SIMO_BUCK_OUT1_VOLTAGE_1_1V);`

Cache Support	RUN_FROM_FLASH
	RUN_FROM_CACHE
		Defines if cache is used. If cache is disabled, the following instructions are used:
		`//Disable CM4 cache`
		`CY_SET_REG32(CYREG_FLASHC_CM4_CA_CTL0, CY_GET_REG32(CYREG_FLASHC_CM4_CA_CTL0) & CACHE_DISABLE_MASK);`
		`//Disable CM0+ cache`
		`CY_SET_REG32(CYREG_FLASHC_CM0_CA_CTL0, CY_GET_REG32(CYREG_FLASHC_CM0_CA_CTL0) & CACHE_DISABLE_MASK);`
		The `Cy_SysLib_SetWaitStates()` function is called based on the CACHE support. If disabled, set the maximum allowed frequency for the given System Mode. If CACHE is enabled, set based on the CM4 CPU frequency.

CM4 CPU Mode	CM4_WHILE_LOOP
	CM4_DHRYSTONE
	CM4_SLEEP
	CM4_DEEP_SLEEP
		Defines what the CM4 CPU runs – While (1) loop, Dhrystone algorithm, go to CPU Sleep, or go to CPU Deep Sleep. The following functions are used:
		`while (1); // Runs a forever while loop`
		`dhrystone(); // Runs the Dhrystone instructions`
		`Cy_SysPm_CpuEnterSleep(CY_SYSPM_WAIT_FOR_INTERRUPT); // Sleep`
		`Cy_SysPm_CpuEnterDeepSleep(CY_SYSPM_WAIT_FOR_INTERRUPT); // Deep-Sleep`

CM4 CPU Frequency [CM4_FREQ_MHZ]	FREQ_8_MHZ
	FREQ_25_MHZ
	FREQ_50_MHZ
	FREQ_100_MHZ
		Defines the clock frequency for the HFCLK0, which is linked to the CM4 CPU clock. The following functions are used:
		`Cy_SysClk_FllConfigure(FREQ_8_MHZ, CM4_FREQ_MHZ, CY_SYSCLK_FLLPLL_OUTPUT_AUTO); // Configure the FLL`
		`Cy_SysClk_FllEnable(TIMEOUT_LOCK); // Enable the FLL`

CM0+ CPU Mode	CM0P_WHILE_LOOP CM0P_DHRYSTONE CM0P_SLEEP CM0P_DEEP_SLEEP CM0P_HIBERNATE	Defines what the CM0+ CPU runs – While (1) loop, Dhrystone algorithm, go to CPU Sleep, go to CPU Deep Sleep or go to hibernate Same functions from CM4 CPU Mode are used. `Cy_SysPm_SystemEnterHibernate(); // Go to Hibernate`

CM0+ CPU Frequency [CM0P_FREQ_MHZ]	FREQ_8_MHZ
	FREQ_25_MHZ
	FREQ_50_MHZ
	FREQ_100_MHZ
		Defines the clock frequency for the CM0+ CPU clock. The following function is used:
		`/* Set the PERI Clock Divider */`
		`Cy_SysClk_ClkPeriSetDivider((CM4_FREQ_MHZ/CM0P_FREQ_MHZ)-1);`

Clock Source	USE_IMO
	USE_FLL
		Defines the clock source for the HCLK0 – IMO or FLL. If IMO is used, the FLL is bypassed. If the FLL is used, the IMO is still used as the source for the FLL.
		The following functions are used:
		`Cy_SysClk_ClkPathSetSource(0UL, CY_SYSCLK_CLKPATH_IN_IMO);`
		`Cy_SysClk_ClkHfSetSource(0UL, CY_SYSCLK_CLKHF_IN_CLKPATH0);`

Minimum Regulator Current Mode	MIN_CURRENT	Determines whether the firmware calls the `Cy_SysPm_SystemSetMinRegulatorCurrent()` function.

For all System LP configurations, there are benchmarks for the BUCK and LDO as the core power supply: `#define VCCD_SUPPLY` sets which type of regulator to use. Note that all the System ULP configurations use only the BUCK. Refer to the System Power Management (SysPm) PDL documentation for more information about the System LP and ULP modes and BUCK and LDO Core voltage regulators.

For all System LP and ULP configurations, there are benchmarks for 3.3 V and 1.8 V device power supplies. There is no option in the firmware to set the power supply voltage. You should manually set the voltage in the kit's hardware.

Note that the compiler optimization level shall be set to *None* to achieve the current measurements listed in the datasheet.

Components and Settings

No components are used in this example.

Reusing This Example

This example is designed for the supported kits. To port the design to a different PSoC 6 MCU device and/or kit, change the target device using the Device Selector.

You can also set the #define SPEC_ID to CUSTOM and edit your own settings in the *spec.h* file. Refer to Table 1 to set the desired configuration.

Related Documents

For a comprehensive list of PSoC 6 MCU resources, see KBA223067 in the Cypress community.

Application Notes	
AN210781 – Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity	Describes PSoC 6 MCU with BLE Connectivity devices and how to build your first PSoC Creator project
AN221774 – Getting Started with PSoC 6 MCU	Describes PSoC 6 MCU devices and how to build your first PSoC Creator 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
AN219434 – Importing PSoC Creator Code into an IDE for a PSoC 6 MCU Project	Describes how to import the code generated by PSoC Creator into your preferred IDE
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 or our GitHub site for a comprehensive collection of code examples using PSoC Creator IDE.	
Device Documentation	
PSoC 6 MCU Datasheets	PSoC 6 Technical Reference Manuals
Development Kit Documentation	
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit	
CY8CKIT-062-WiFi-BT PSoC 6 WiFi-BT Pioneer Kit	
CY8CPROTO-062-4343W PSoC 6 Wi-Fi BT Prototyping Kit	
CY8CPROTO-063 BLE PSoC 6 BLE Prototyping Kit	
Tool Documentation	
PSoC Creator	Look in the downloads tab for Quick Start and User Guides
Peripheral Driver Library (PDL)	Get the latest version for use with PSoC Creator. Look in the *<PDL install folder>/doc* for the User Guide and the API Reference

Document History

Document Title: CE226306 – PSoC 6 MCU Power Measurements

Document Number: 002-26306

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6472014	RLOS	02/01/2019	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 | Code Examples | Projects | Videos | Blogs | Training| Components

Technical Support

cypress.com/support

All other trademarks or registered trademarks referenced herein are the property of their respective owners.

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

© Cypress Semiconductor Corporation, 2019. This document is the property of Cypress Semiconductor Corporation and its subsidiaries ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. No computing device can be absolutely secure. Therefore, despite security measures implemented in Cypress hardware or software products, Cypress shall have no liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. CYPRESS DOES NOT REPRESENT, WARRANT, OR GUARANTEE THAT CYPRESS PRODUCTS, OR SYSTEMS CREATED USING CYPRESS PRODUCTS, WILL BE FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (collectively, "Security Breach"). Cypress disclaims any liability relating to any Security Breach, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from any Security Breach. In addition, the products described in these materials may contain design defects or errors known as errata which may cause the product to deviate from published specifications. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. "High-Risk Device" means any device or system whose failure could cause personal injury, death, or property damage. Examples of High-Risk Devices are weapons, nuclear installations, surgical implants, and other medical devices. "Critical Component" means any component of a High-Risk Device whose failure to perform can be reasonably expected to cause, directly or indirectly, the failure of the High-Risk Device, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from any use of a Cypress product as a Critical Component in a High-Risk Device. You shall indemnify and hold Cypress, its directors, officers, employees, agents, affiliates, distributors, and assigns harmless from and against all claims, costs, damages, and expenses, arising out of any claim, including claims for product liability, personal injury or death, or property damage arising from any use of a Cypress product as a Critical Component in a High-Risk Device. Cypress products are not intended or authorized for use as a Critical Component in any High-Risk Device except to the limited extent that (i) Cypress's published data sheet for the product explicitly states Cypress has qualified the product for use in a specific High-Risk Device, or (ii) Cypress has given you advance written authorization to use the product as a Critical Component in the specific High-Risk Device and you have signed a separate indemnification agreement.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.