

CY8CKIT-042

PSoC[®] 4 Pioneer Kit Guide

Doc. # 001-86371 Rev. *G

Cypress Semiconductor 198 Champion Court San Jose, CA 95134-1709 Phone (USA): 800.858.1810 Phone (Intnl): +1.408.943.2600 http://www.cypress.com



Copyrights

© Cypress Semiconductor Corporation, 2013-2016. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("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 PUR-POSE. 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. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, 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 or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities. including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, 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.

PSoC Designer, PSoC Creator, SmartSense, and CapSense Express are trademarks of Cypress Semiconductor Corporation.

Contents



Safety Information 5

1.	Introdu	ction	7
	1.1	Kit Contents	7
	1.2	PSoC Creator™	
	1.3	Getting Started	
	1.4	Additional Learning Resources	
		1.4.1 PSoC Creator	
		1.4.2 PSoC Creator Code Examples	
		1.4.3 PSoC Creator Help	
		1.4.4 Technical Support	
	1.5	Documentation Conventions	14
2.	Softwa	re Installation	15
	2.1	Install Kit Software	15
	2.2	Install Hardware	16
	2.3	Install Software	
	2.4	Uninstall Software	
	2.5	Develop Code Fast and Easy with Code Examples	
	2.6	Open an Example Project in PSoC Creator	19
3.	Kit Ope	eration	21
			~~
	3.1	Pioneer Kit USB Connection	
	3.1 3.2	Pioneer Kit USB Connection Programming and Debugging PSoC 4	
		Programming and Debugging PSoC 4	23
		Programming and Debugging PSoC 4	23 23
		Programming and Debugging PSoC 4 3.2.1 Using the Onboard PSoC 5LP Programmer and Debugger	23 23 25
	3.2	Programming and Debugging PSoC 4 3.2.1 Using the Onboard PSoC 5LP Programmer and Debugger 3.2.2 Using CY8CKIT-002 MiniProg3 Programmer and Debugger USB-UART Bridge USB-I2C Bridge	23 23 25 26 28
	3.2	Programming and Debugging PSoC 4 3.2.1 Using the Onboard PSoC 5LP Programmer and Debugger 3.2.2 Using CY8CKIT-002 MiniProg3 Programmer and Debugger USB-UART Bridge	23 23 25 26 28
4.	3.2 3.3 3.4	Programming and Debugging PSoC 4 3.2.1 Using the Onboard PSoC 5LP Programmer and Debugger 3.2.2 Using CY8CKIT-002 MiniProg3 Programmer and Debugger USB-UART Bridge USB-I2C Bridge Updating the Onboard Programmer Firmware	23 23 25 26 28
4.	3.2 3.3 3.4 3.5	Programming and Debugging PSoC 4 3.2.1 Using the Onboard PSoC 5LP Programmer and Debugger 3.2.2 Using CY8CKIT-002 MiniProg3 Programmer and Debugger USB-UART Bridge USB-I2C Bridge Updating the Onboard Programmer Firmware	23 25 26 28 29 31
4.	3.2 3.3 3.4 3.5 Hardwa	Programming and Debugging PSoC 4 3.2.1 Using the Onboard PSoC 5LP Programmer and Debugger 3.2.2 Using CY8CKIT-002 MiniProg3 Programmer and Debugger USB-UART Bridge USB-I2C Bridge Updating the Onboard Programmer Firmware	23 25 26 28 29 31 31
4.	3.2 3.3 3.4 3.5 Hardwa 4.1	Programming and Debugging PSoC 4 3.2.1 Using the Onboard PSoC 5LP Programmer and Debugger 3.2.2 Using CY8CKIT-002 MiniProg3 Programmer and Debugger USB-UART Bridge USB-I2C Bridge Updating the Onboard Programmer Firmware	23 25 26 28 29 31 31 33
4.	3.2 3.3 3.4 3.5 Hardwa 4.1 4.2	Programming and Debugging PSoC 4 3.2.1 Using the Onboard PSoC 5LP Programmer and Debugger 3.2.2 Using CY8CKIT-002 MiniProg3 Programmer and Debugger USB-UART Bridge USB-I2C Bridge Updating the Onboard Programmer Firmware Ire Board Details Theory of Operation Functional Description 4.3.1 PSoC 4	23 25 26 28 29 31 31 33 34 34
4.	3.2 3.3 3.4 3.5 Hardwa 4.1 4.2	Programming and Debugging PSoC 4 3.2.1 Using the Onboard PSoC 5LP Programmer and Debugger 3.2.2 Using CY8CKIT-002 MiniProg3 Programmer and Debugger USB-UART Bridge USB-I2C Bridge Updating the Onboard Programmer Firmware Intervet and Details Theory of Operation Functional Description 4.3.1 PSoC 4 4.3.2 PSoC 5LP	23 25 26 28 29 31 31 31 34 34 35
4.	3.2 3.3 3.4 3.5 Hardwa 4.1 4.2	Programming and Debugging PSoC 4 3.2.1 Using the Onboard PSoC 5LP Programmer and Debugger 3.2.2 Using CY8CKIT-002 MiniProg3 Programmer and Debugger USB-UART Bridge USB-I2C Bridge Updating the Onboard Programmer Firmware Ire Board Details Theory of Operation Functional Description 4.3.1 PSoC 4 4.3.2 PSoC 5LP 4.3.3 Power Supply System	23 25 26 28 29 31 31 33 34 34 35 37
4.	3.2 3.3 3.4 3.5 Hardwa 4.1 4.2	Programming and Debugging PSoC 4 3.2.1 Using the Onboard PSoC 5LP Programmer and Debugger 3.2.2 Using CY8CKIT-002 MiniProg3 Programmer and Debugger USB-UART Bridge USB-I2C Bridge Updating the Onboard Programmer Firmware Inter Board Details Theory of Operation Functional Description 4.3.1 PSoC 4 4.3.2 PSoC 5LP 4.3.3 Power Supply System 4.3.4 Programming Interface	23 25 26 28 29 31 31 33 34 34 35 37 39
4.	3.2 3.3 3.4 3.5 Hardwa 4.1 4.2	Programming and Debugging PSoC 4 3.2.1 Using the Onboard PSoC 5LP Programmer and Debugger 3.2.2 Using CY8CKIT-002 MiniProg3 Programmer and Debugger USB-UART Bridge USB-I2C Bridge Updating the Onboard Programmer Firmware Infe Board Details Theory of Operation Functional Description 4.3.1 PSoC 4 4.3.2 PSoC 5LP 4.3.3 Power Supply System 4.3.4 Programming Interface. 4.3.5 Arduino Compatible Headers (J1, J2, J3, J4, and J12 - unpopulated)	23 25 26 28 29 31 31 33 34 34 35 37 39 40
4.	3.2 3.3 3.4 3.5 Hardwa 4.1 4.2	Programming and Debugging PSoC 4 3.2.1 Using the Onboard PSoC 5LP Programmer and Debugger 3.2.2 Using CY8CKIT-002 MiniProg3 Programmer and Debugger USB-UART Bridge USB-I2C Bridge Updating the Onboard Programmer Firmware Ire Board Details Theory of Operation Functional Description 4.3.1 PSoC 4 4.3.2 PSoC 5LP 4.3.3 Power Supply System 4.3.4 Programming Interface 4.3.5 Arduino Compatible Headers (J1, J2, J3, J4, and J12 - unpopulated) 4.3.6 Digilent Pmod Compatible Header (J5 - unpopulated)	23 25 26 28 29 31 31 33 34 34 35 37 39 40 42
4.	3.2 3.3 3.4 3.5 Hardwa 4.1 4.2	Programming and Debugging PSoC 4 3.2.1 Using the Onboard PSoC 5LP Programmer and Debugger 3.2.2 Using CY8CKIT-002 MiniProg3 Programmer and Debugger USB-UART Bridge USB-I2C Bridge Updating the Onboard Programmer Firmware Infe Board Details Theory of Operation Functional Description 4.3.1 PSoC 4 4.3.2 PSoC 5LP 4.3.3 Power Supply System 4.3.4 Programming Interface. 4.3.5 Arduino Compatible Headers (J1, J2, J3, J4, and J12 - unpopulated)	23 25 26 28 29 31 31 31 34 34 34 35 37 39 40 42 43



		4.3.9 Pioneer Board LEDs	45
		4.3.10 Push Buttons	46
5.	Code E	xamples	47
	5.1	Project: Blinking LED	
	••••	5.1.1 Project Description	
		5.1.2 Hardware Connections	
		5.1.3 Flow Chart	
		5.1.4 Verify Output	
	5.2	Project: PWM	
		5.2.1 Project Description	52
		5.2.2 Hardware Connections	
		5.2.3 Flow Chart	53
		5.2.4 Verify Output	53
	5.3	Project: Deep Sleep	54
		5.3.1 Project Description	54
		5.3.2 Hardware Connections	54
		5.3.3 Flow Chart	55
		5.3.4 Verify Output	55
	5.4	Project: CapSense	56
		5.4.1 CapSense (Without Tuning)	56
		5.4.2 CapSense (With Tuning)	59
6.	Advand	ed Topics	66
6.		•	
6.	6.1	Using PSoC 5LP as USB-UART Bridge	66
6.	6.1 6.2	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge	66 79
6.	6.1	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP	66 79 88
6.	6.1 6.2	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP 6.3.1 Building a Bootloadable Project for PSoC 5LP	
6.	6.1 6.2 6.3	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP 6.3.1 Building a Bootloadable Project for PSoC 5LP 6.3.2 Building a Normal Project for PSoC 5LP	
6.	6.1 6.2	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP 6.3.1 Building a Bootloadable Project for PSoC 5LP 6.3.2 Building a Normal Project for PSoC 5LP PSoC 5LP Factory Program Restore Instructions	
6.	6.1 6.2 6.3	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP 6.3.1 Building a Bootloadable Project for PSoC 5LP 6.3.2 Building a Normal Project for PSoC 5LP PSoC 5LP Factory Program Restore Instructions 6.4.1 PSoC 5LP is Programmed with a Bootloadable Application	
6.	6.1 6.2 6.3 6.4	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP 6.3.1 Building a Bootloadable Project for PSoC 5LP 6.3.2 Building a Normal Project for PSoC 5LP PSoC 5LP Factory Program Restore Instructions 6.4.1 PSoC 5LP is Programmed with a Bootloadable Application 6.4.2 PSoC 5LP is Programmed with a Standard Application	
6.	6.1 6.2 6.3	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP 6.3.1 Building a Bootloadable Project for PSoC 5LP 6.3.2 Building a Normal Project for PSoC 5LP PSoC 5LP Factory Program Restore Instructions 6.4.1 PSoC 5LP is Programmed with a Bootloadable Application 6.4.2 PSoC 5LP is Programmed with a Standard Application Using µC/Probe Tool	
6.	6.1 6.2 6.3 6.4	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP 6.3.1 Building a Bootloadable Project for PSoC 5LP 6.3.2 Building a Normal Project for PSoC 5LP PSoC 5LP Factory Program Restore Instructions 6.4.1 PSoC 5LP is Programmed with a Bootloadable Application 6.4.2 PSoC 5LP is Programmed with a Standard Application Using µC/Probe Tool	
	6.1 6.2 6.3 6.4 6.5	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP 6.3.1 Building a Bootloadable Project for PSoC 5LP 6.3.2 Building a Normal Project for PSoC 5LP PSoC 5LP Factory Program Restore Instructions 6.4.1 PSoC 5LP is Programmed with a Bootloadable Application 6.4.2 PSoC 5LP is Programmed with a Standard Application Using μC/Probe Tool 6.5.1 CapSense Example Project 6.5.2 PWM Example project	
	6.1 6.2 6.3 6.4 6.5	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP 6.3.1 Building a Bootloadable Project for PSoC 5LP 6.3.2 Building a Normal Project for PSoC 5LP PSoC 5LP Factory Program Restore Instructions 6.4.1 PSoC 5LP is Programmed with a Bootloadable Application 6.4.2 PSoC 5LP is Programmed with a Standard Application Using μ C/Probe Tool 6.5.1 CapSense Example Project 6.5.2 PWM Example project	
	6.1 6.2 6.3 6.4 6.5 Appenc A.1	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP 6.3.1 Building a Bootloadable Project for PSoC 5LP 6.3.2 Building a Normal Project for PSoC 5LP PSoC 5LP Factory Program Restore Instructions 6.4.1 PSoC 5LP is Programmed with a Bootloadable Application 6.4.2 PSoC 5LP is Programmed with a Standard Application Using μC/Probe Tool 6.5.1 CapSense Example Project 6.5.2 PWM Example project lix CY8CKIT-042 Schematics	
	6.1 6.2 6.3 6.4 6.5 Appenc A.1 A.2	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP	
	6.1 6.2 6.3 6.4 6.5 Appenc A.1 A.2 A.3	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP 6.3.1 Building a Bootloadable Project for PSoC 5LP 6.3.2 Building a Normal Project for PSoC 5LP PSoC 5LP Factory Program Restore Instructions 6.4.1 PSoC 5LP is Programmed with a Bootloadable Application 6.4.2 PSoC 5LP is Programmed with a Standard Application Using μC/Probe Tool 6.5.1 CapSense Example Project 6.5.2 PWM Example project bitx CY8CKIT-042 Schematics Pin Assignment Table Program and Debug Headers	
	6.1 6.2 6.3 6.4 6.5 Append A.1 A.2 A.3 A.4	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP 6.3.1 Building a Bootloadable Project for PSoC 5LP 6.3.2 Building a Normal Project for PSoC 5LP PSoC 5LP Factory Program Restore Instructions 6.4.1 PSoC 5LP is Programmed with a Bootloadable Application 6.4.2 PSoC 5LP is Programmed with a Standard Application Using μC/Probe Tool 6.5.1 CapSense Example Project 6.5.2 PWM Example project Iix CY8CKIT-042 Schematics Pin Assignment Table Program and Debug Headers Use of Zero-ohm Resistors and No Load	
	6.1 6.2 6.3 6.4 6.5 Appenc A.1 A.2 A.3 A.4 A.5	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP 6.3.1 Building a Bootloadable Project for PSoC 5LP 6.3.2 Building a Normal Project for PSoC 5LP PSoC 5LP Factory Program Restore Instructions 6.4.1 PSoC 5LP is Programmed with a Bootloadable Application 6.4.2 PSoC 5LP is Programmed with a Standard Application Using μC/Probe Tool 6.5.1 CapSense Example Project 6.5.2 PWM Example project 9.5.2 Program and Debug Headers 9.5.2 Use of Zero-ohm Resistors and No Load 1.5.4 Error in Firmware/Status Indication in Status LED 1.5.4	
	6.1 6.2 6.3 6.4 6.5 Appenc A.1 A.2 A.3 A.4 A.5 A.6	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP 6.3.1 Building a Bootloadable Project for PSoC 5LP 6.3.2 Building a Normal Project for PSoC 5LP PSoC 5LP Factory Program Restore Instructions 6.4.1 PSoC 5LP is Programmed with a Bootloadable Application 6.4.2 PSoC 5LP is Programmed with a Standard Application Using μC/Probe Tool 6.5.1 CapSense Example Project 6.5.2 PWM Example project 6.5.2 Pin Assignment Table Program and Debug Headers Use of Zero-ohm Resistors and No Load Error in Firmware/Status Indication in Status LED Bill of Materials (BOM) Status Indication in Status LED	
	6.1 6.2 6.3 6.4 6.5 Appenc A.1 A.2 A.3 A.4 A.5	Using PSoC 5LP as USB-UART Bridge Using PSoC 5LP as USB-I2C Bridge Developing Applications for PSoC 5LP 6.3.1 Building a Bootloadable Project for PSoC 5LP 6.3.2 Building a Normal Project for PSoC 5LP PSoC 5LP Factory Program Restore Instructions 6.4.1 PSoC 5LP is Programmed with a Bootloadable Application 6.4.2 PSoC 5LP is Programmed with a Standard Application Using μC/Probe Tool 6.5.1 CapSense Example Project 6.5.2 PWM Example project 9.5.2 Program and Debug Headers 9.5.2 Use of Zero-ohm Resistors and No Load 1.5.4 Error in Firmware/Status Indication in Status LED 1.5.4	

Revision History

128

Safety Information



Regulatory Compliance

The CY8CKIT-042 PSoC[®] 4 Pioneer Kit is intended for use as a development platform for hardware or software in a laboratory environment. The board is an open system design, which does not include a shielded enclosure. Due to this reason, the board may cause interference to other electrical or electronic devices in close proximity. In a domestic environment, this product may cause radio interference. In such cases, the user may be required to take adequate preventive measures. Also, this board should not be used near any medical equipment or RF devices.

Attaching additional wiring to this product or modifying the product operation from the factory default may affect its performance and cause interference with other apparatus in the immediate vicinity. If such interference is detected, suitable mitigating measures should be taken.

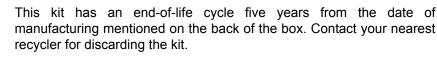
The CY8CKIT-042 as shipped from the factory has been verified to meet with requirements of CE as a Class A product.





The CY8CKIT-042 contains electrostatic discharge (ESD) sensitive devices. Electrostatic charges readily accumulate on the human body and any equipment, and can discharge without detection. Permanent damage may occur on devices subjected to high-energy discharges. Proper ESD precautions are recommended to avoid performance degradation or loss of functionality. Store unused CY8CKIT-042 boards in the protective shipping package.

End-of-Life/Product Recycling





General Safety Instructions

ESD Protection

ESD can damage boards and associated components. Cypress recommends that the user perform procedures only at an ESD workstation. If an ESD workstation is not available, use appropriate ESD protection by wearing an antistatic wrist strap attached to the chassis ground (any unpainted metal surface) on the board when handling parts.

Handling Boards

CY8CKIT-042 boards are sensitive to ESD. Hold the board only by its edges. After removing the board from its box, place it on a grounded, static free surface. Use a conductive foam pad if available. Do not slide board over any surface.



Thank you for your interest in the PSoC[®] 4 Pioneer Kit. The kit is designed as an easy-to-use and inexpensive development kit, showcasing the unique flexibility of the PSoC 4 architecture. Designed for flexibility, this kit offers footprint-compatibility with several third-party Arduino[™] shields. This kit has a provision to populate an extra header to support Digilent[®] Pmod[™] peripheral modules. In addition, the board features a CapSense[®] slider, an RGB LED, a push button switch, an integrated USB programmer, a program and debug header, and USB-UART/I2C bridges. This kit supports either 5 V or 3.3 V as power supply voltages.

The PSoC 4 Pioneer Kit is based on the PSoC 4200 device family, delivering a programmable platform for a wide range of embedded applications. The PSoC 4 is a scalable and reconfigurable platform architecture for a family of mixed-signal programmable embedded system controllers with an ARM[®] Cortex[™]-M0 CPU. It combines programmable and reconfigurable analog and digital blocks with flexible automatic routing.

1.1 Kit Contents

The PSoC 4 Pioneer kit contains:

- PSoC 4 Pioneer board
- Quick Start Guide
- USB Standard-A to Mini-B cable
- Six jumper wires



Figure 1-1. Kit Contents



Inspect the contents of the kit; if you find any part missing, contact your nearest Cypress sales office for help: www.cypress.com/go/support.



1.2 PSoC Creator[™]

PSoC Creator is a state-of-the-art, easy-to-use integrated design environment (IDE). It introduces revolutionary hardware and software co-design, powered by a library of pre-verified and pre-characterized PSoC Components[™].

With PSoC Creator, you can:

- Drag and drop PSoC components to build a schematic of your custom design
- Automatically place and route components and configure GPIOs
- Develop and debug firmware using the included component APIs

PSoC Creator also enables you to tap into an entire tools ecosystem with integrated compiler chains and production programmers for PSoC devices.

For more information, visit www.cypress.com/Creator.

1.3 Getting Started

This guide helps you to get acquainted with the PSoC 4 Pioneer Kit. The Software Installation chapter on page 15 describes the installation of the kit software. The Kit Operation chapter on page 21 explains how to program the PSoC 4 with a programmer and debugger – either the onboard PSoC 5LP or the external MiniProg3 (CY8CKIT-002). The Hardware chapter on page 31 details the hardware operation. The Code Examples chapter on page 47 describes the code examples. The Advanced Topics chapter on page 66 deals with topics such as building projects for PSoC 5LP, USB-UART functionality, and USB-I2C functionality of PSoC 5LP. The Appendix on page 113 provides the schematics, pin assignment, use of zero-ohm resistors, troubleshooting, and the bill of materials (BOM).



1.4 Additional Learning Resources

Cypress provides a wealth of data at www.cypress.com to help you to select the right PSoC device for your design, and to help you to quickly and effectively integrate the device into your design. For a comprehensive list of resources, see KBA86521, How to Design with PSoC 3, PSoC 4, and PSoC 5LP. The following is an abbreviated list for PSoC 4:

- Overview: PSoC Portfolio, PSoC Roadmap
- Product Selectors: PSoC 1, PSoC 3, PSoC 4, or PSoC 5LP. In addition, PSoC Creator includes a device selection tool.
- Datasheets: Describe and provide electrical specifications for the PSoC 4000, PSoC 4100, and PSoC 4200 device families.
- CapSense Design Guide: Learn how to design capacitive touch-sensing applications with the PSoC 4 family of devices.
- Application Notes and Code Examples: Cover a broad range of topics, from basic to advanced level. Many of the application notes include code examples. Visit the PSoC 3/4/5 Code Examples webpage for a list of all available PSoC Creator code examples. For accessing code examples from within PSoC Creator see PSoC Creator Code Examples on page 12.
- Technical Reference Manuals (TRM): Provide detailed descriptions of the architecture and registers in each PSoC 4 device family.
- Development Kits:
 - CY8CKIT-042 and CY8CKIT-040, PSoC 4 Pioneer Kits, are easy-to-use and inexpensive development platforms. These kits include connectors for Arduino compatible shields and Digilent Pmod daughter cards.
 - □ CY8CKIT-049 is a very low-cost prototyping platform for sampling PSoC 4 devices.
 - □ CY8CKIT-001 is a common development platform for all PSoC family devices.
- The MiniProg3 device provides an interface for flash programming and debug.
- Knowledge Base Articles (KBA): Provide design and application tips from experts on the devices/kits. For instance, KBA93541, explains how to use CY8CKIT-049 to program another PSoC 4.

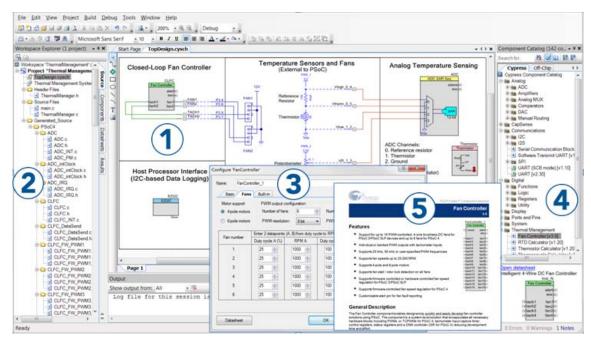


1.4.1 PSoC Creator

PSoC Creator is a free Windows-based integrated design environment (IDE). It enables concurrent hardware and firmware design of systems based on PSoC 3, PSoC 4, and PSoC 5LP. See Figure 1-2 – with PSoC Creator, you can:

- 1. Drag and drop Components to build your hardware system design in the main design workspace
- 2. Codesign your application firmware with the PSoC hardware
- 3. Configure Components using configuration tools
- 4. Explore the library of 100+ Components
- 5. Access Component datasheets

Figure 1-2. PSoC Creator Features



Visit PSoC Creator training page for video tutorials on learning and using PSoC Creator.



1.4.2 PSoC Creator Code Examples

PSoC Creator includes a large number of code example projects. These projects are accessible from the PSoC Creator Start Page, as Figure 1-3 shows.

Example projects can speed up your design process by starting you off with a complete design, instead of a blank page. The example projects also show how PSoC Creator Components can be used for various applications. Code examples and datasheets are included, as Figure 1-4 on page 13 shows.

In the Find Example Project dialog shown in Figure 1-4 on page 13, you have several options:

- Filter for examples based on device family or keyword
- Select from the menu of examples offered based on the Filter Options
- View the datasheet for the selection (on the **Documentation** tab)
- View the code example for the selection. You can copy and paste code from this window to your project, which can help speed up code development, or
- Create a new workspace for the example project. This can speed up your design process by starting you off with a complete, basic design. You can then adapt that design to your application.

Figure 1-3. Code Examples in PSoC Creator

Start Page	
PSoC [®] Creator [™]	
Recent Projects	
Design01.cywrk PSoC5LP_CSP_Btldr.cywrk PSoC5LP_CSP_Btldr.cywrk PSoC5LP_CSP_Btldr.cywrk PSoC3_CSP_Btldr.cywrk I1.cywrk	
Create New Project Open Existing Project	
Getting Started	
PSoC Creator Start Page	
Quick Start Guide	
Intro to PSoC	
Intro to PSoC Creator	
PSoC Creator Training	
Design Tutorials	-
Getting Started With PSoC 3	
Getting Started With PSoC 4	/
Getting Started With PSoC 5LP	
Examples and Kits	
Find Example Project	
🗉 Kita 🧭	
Product Information	
PSoC Creator	
PSoC Programmer	
PSoC 3	
PSoC 4	
PSoC 5LP	
Resources	
Cypress Dev Community	
Application Notes PSoCDeveloper.com	12



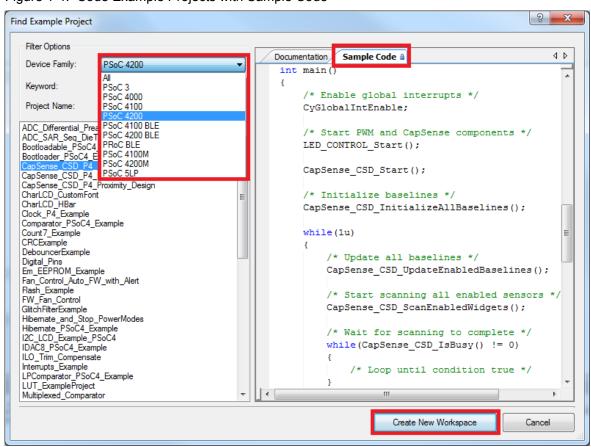


Figure 1-4. Code Example Projects with Sample Code

1.4.3 PSoC Creator Help

Visit the PSoC Creator home page to download the latest version of PSoC Creator. Then, launch PSoC Creator and navigate to the following items:

- Quick Start Guide: Choose Help > Documentation > Quick Start Guide. This guide gives you the basics for developing PSoC Creator projects.
- Simple Component example projects: Choose File > Open > Example projects. These example projects demonstrate how to configure and use PSoC Creator Components.
- Starter designs: Choose File > New > Project > PSoC 4100 / PSoC 4200 Starter Designs. These starter designs demonstrate the unique features of PSoC 4.
- System Reference Guide: Choose Help > System Reference > System Reference Guide. This guide lists and describes the system functions provided by PSoC Creator.
- Component datasheets: Right-click a Component and select "Open Datasheet." Visit the PSoC 4 Component Datasheets page for a list of all PSoC 4 Component datasheets.
- Document Manager: PSoC Creator provides a document manager to help you to easily find and review document resources. To open the document manager, choose the menu item Help > Document Manager.



1.4.4 Technical Support

If you have any questions, our technical support team is happy to assist you. You can create a support request on the Cypress Technical Support page.

If you are in the United States, you can talk to our technical support team by calling our toll-free number: +1-800-541-4736. Select option 2 at the prompt.

You can also use the following support resources if you need quick assistance.

- Self-help
- Local Sales Office Locations

1.5 Documentation Conventions

Table 1-1. Document Conventions for Guides		nventions for Guides
	Convention	

Convention	Usage
Courier New	Displays file locations, user entered text, and source code: C:\cd\icc\
Italics	Displays file names and reference documentation: Read about the <i>sourcefile.hex</i> file in the <i>PSoC Creator User Guide</i> .
[Bracketed, Bold]	Displays keyboard commands in procedures: [Enter] or [Ctrl] [C]
File > Open	Represents menu paths: File > Open > New Project
Bold	Displays commands, menu paths, and icon names in procedures: Click the File icon and then click Open .
Times New Roman	Displays an equation: 2+2=4
Text in gray boxes	Describes cautions or unique functionality of the product.

2. Software Installation



2.1 Install Kit Software

Follow these steps to install the PSoC 4 Pioneer Kit software:

- 1. Download and install the PSoC 4 Pioneer Kit software from www.cypress.com/go/CY8CKIT-042.
- 2. Select the folder to install the CY8CKIT-042 related files. Choose the directory and click Next.

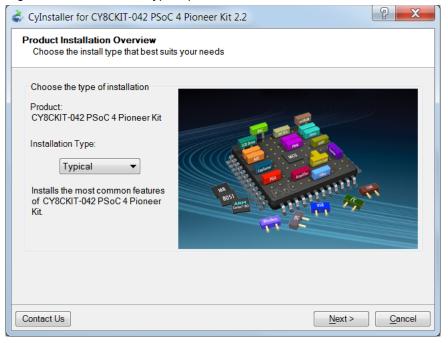
Figure 2-1. Installation Folder

CY8CKIT-042 PSoC 4 Pioneer Kit -	InstallShield Wizard
	Welcome to the InstallShield Wizard for CY8CKIT-042 PSoC 4 Pioneer Kit The InstallShield Wizard will install CY8CKIT-042 PSoC 4 Pioneer Kit on your computer. To continue, click Next.
	Select folder where setup will install files. Install CY8CKIT-042 PSoC 4 Pioneer Kit C:\\Cypress
	< Back Next > Cancel



3. Select the installation type and click Next.

Figure 2-2. Installation Type Options



4. Read and accept the End-User Licence Agreement and click **Next** to proceed with the installation.

After the installation is complete, the kit contents are available at the following location: <Install_Directory>\CY8CKIT-042 PSoC 4 Pioneer Kit\<version>

Note For Windows 7 users, the installed files and the folder are read-only. To change the property, right-click the folder and select **Properties > Attributes**; disable the **Read-only** radio button. Click **Apply** and **OK** to close the window.

2.2 Install Hardware

There is no additional hardware installation required for this kit.

2.3 Install Software

When installing the PSoC 4 Pioneer Kit, the installer checks if the required software is installed in the system. If the required applications are not installed, then the installer prompts you to download and install them.

The following software is required:

- PSoC Creator 3.2 Service Pack 1 or later: Download the latest software from www.cypress.com/go/Creator.
- PSoC Programmer 3.23.1 or later: Download the latest software from www.cypress.com/go/Programmer.
- Code examples: After the kit installation is complete, the code examples are available in the kit firmware folder. Download the CD ISO image or the setup files to install the kit from www.cypress.com/go/CY8CKIT-042.



2.4 Uninstall Software

The software can be uninstalled using one of the following methods:

- Go to Start > All Programs > Cypress > Cypress Update Manager > Cypress Update Manager; select the Uninstall button.
- Go to Start > Control Panel > Programs and Features; select the Uninstall/Change button.

2.5 Develop Code Fast and Easy with Code Examples

PSoC Creator provides several example projects that make code development fast and easy. To access these projects, click **Find Example Project...** under the **Example and Kits** section in the **Start Page** of PSoC Creator or navigate to the Creator tool bar and select **File > Example Project**.

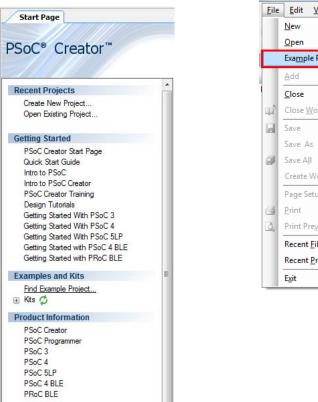
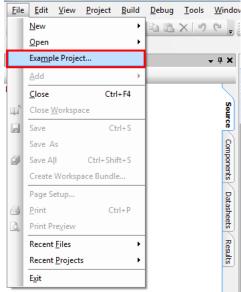


Figure 2-3. Find Example Project



The Find Example Project section has various filters that help to locate the most relevant project.

PSoC Creator also provides several starter designs for each device family. These designs highlight features that are unique to each PSoC family. They provide users with a starting place instead of creating a new empty design. These starter projects come loaded with various pre-selected components. To use a starter design, navigate to **File > New > Project** and select the design required.

New Project				
Design Other			<	1 Þ
 Default Templates 				
📃 PSoC 3 Desi	gn	Creates a PSoC 3, 8-bit 8051, design project.		
PSoC 4000	Design	Creates a PSoC 4000, 32-bit ARM Cortex-M0, design project.		
PSoC 4100 /	/ PSoC 4200 Design	Creates a PSoC 4100 / PSoC 4200, 32-bit ARM Cortex-M0, design project.		
PSoC 4100	BLE / PSoC 4200 BLE Design	Creates a PSoC 4100 BLE / PSoC 4200 BLE, 32-bit ARM Cortex-M0, design project		=
PRoC BLE D	esign	Creates a PRoC BLE, 32-bit ARM Cortex-M0, design project.		
PSoC 4100N	M / PSoC 4200M Design	Creates a PSoC 4100M / PSoC 4200M, 32-bit ARM Cortex-M0, design project.		
PSoC 5LP D	esign	Creates a PSoC 5LP, 32-bit ARM Cortex-M3, design project.		
PSoC 3 Starter Designation	gns			
ADC_DMA_	VDAC	Shows how to transfer data from an ADC to a DAC using DMA with no CPU intervention.		
▶ DelSig_16Channel		Shows a 16-channel, 12-bit Delta Sigma ADC sequenced in hardware; samples are transferred from ADC to SRAM using DMA - without processor intervention.	2	
DelSig_I2CM		Shows the 16-bit differential ADC, hardware multiplexed into 8 channels and transported over I2C.		
DelSig_I2CS		Shows the 16-bit differential ADC, hardware multiplexed into 8 channels and transported over I2C.		
Name:	Design01			
Location:	C:\Users\srds\Documents			
Device:	CY8C3866AXI-040 - (Default	PSoC 3 Device)	•	
Workspace: Create New Workspace			-	
Workspace name: Design01				
Project template: Empty schematic			-	
		ОК	Cancel	

Figure 2-4. Starter Designs

In addition to the example projects and starter designs that are available within PSoC Creator, Cypress continuously strives to provide the best support. Click here to view a growing list of application notes for PSoC 3, PSoC 4, and PSoC 5LP.



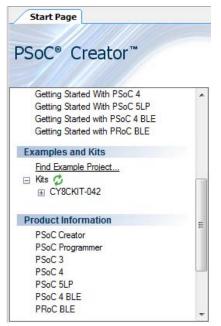
2.6 Open an Example Project in PSoC Creator

1. Launch PSoC Creator from the Start menu.



Open the example project from the Start Page by clicking <Project.cywrk> present under Examples and Kits > Kits > CY8CKIT-042.

Figure 2-6. Open Example Project





3. The example project opens and displays the project files in the Workspace Explorer. Subsequent sections of this user guide describe how to build, program, and understand the example projects supported in this kit.

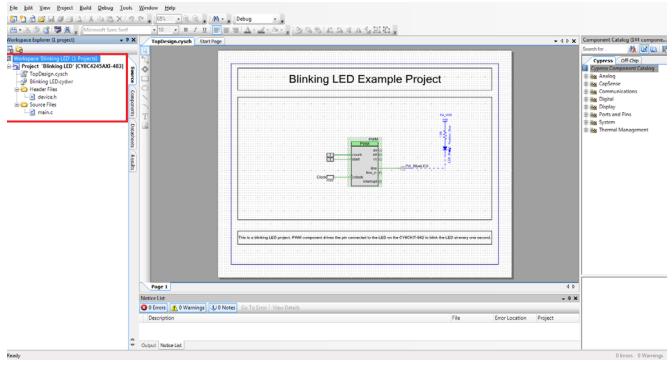


Figure 2-7. Workspace Explorer

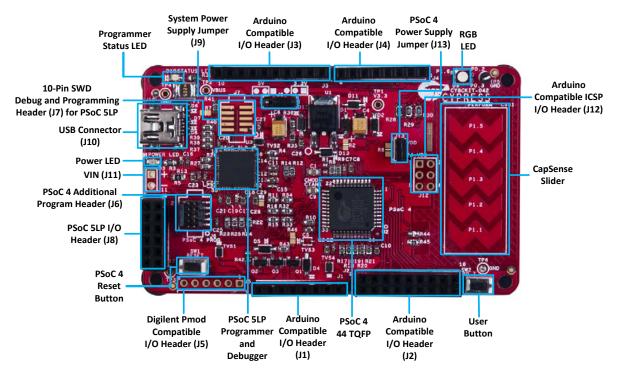


The PSoC 4 Pioneer Kit can be used to develop applications using the PSoC 4 family of devices and the Arduino shields and Digilent Pmod daughter cards. Figure 3-1 is an image of the PSoC 4 Pioneer board with a markup of the onboard components.



Kit Operation

3.





3.1 Pioneer Kit USB Connection

The PSoC 4 Pioneer Kit connects to the PC over a USB interface. The kit enumerates as a composite device and three separate devices appear under the Device Manager window in the Windows operating system.

		-
Table 3-1	PSoC 4 Pioneer Kit in Device Manager A	After Enumeration
	T SOC 4 T IONEEL MILLIN DEVICE Manager F	

Port	Description
USB Composite Device	Composite device
USB Input Device	USB-I ² C bridge, KitProg command interface
KitProg	Programmer and debugger
KitProg USB-UART	USB-UART bridge, which appears as the COM# port

Figure 3-2. KitProg Driver Installation

Driver Software Installation		×
Your device is ready to use		
USB Composite Device USB Input Device KitProg (1.2.3.3) KitProg USB-UART (COM28)	 Ready to use Ready to use Ready to use Ready to use 	
		Close



3.2 **Programming and Debugging PSoC 4**

The kit allows programming and debugging of the PSoC 4 device in two modes:

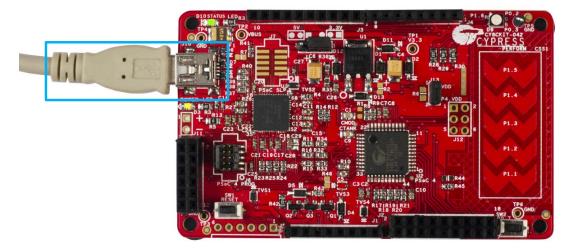
- Using the onboard PSoC 5LP programmer and debugger
- Using a CY8CKIT-002 MiniProg3 programmer and debugger

3.2.1 Using the Onboard PSoC 5LP Programmer and Debugger

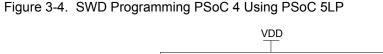
The default programming interface for the kit is a USB-based, onboard programming interface. Before trying to program the device, PSoC Creator and PSoC Programmer must be installed. See Install Software on page 16 for information on installing the kit software.

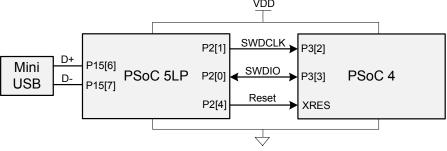
 To program the device, plug the USB cable into the programming USB connector J10, as shown in Figure 3-3. The kit will enumerate as a composite device. See Pioneer Kit USB Connection on page 22 for details.

Figure 3-3. Connect USB Cable to J10



2. The onboard PSoC 5LP uses serial wire debug (SWD) to program the PSoC 4 device. See Figure 3-4 for this implementation.

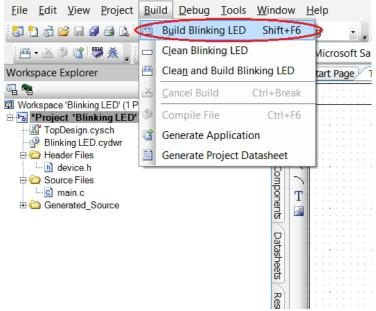






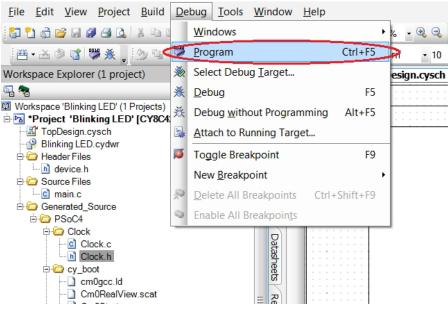
 The Pioneer Kit's onboard programmer will enumerate on the PC and in the software tools as KitProg. Load an example project in PSoC Creator (such as the project described in Install Software on page 16) and initiate the build by clicking Build > Build Project or [Shift]+[F6].

Figure 3-5. Build Project in PSoC Creator



4. After the project is built without errors and warnings, select **Debug > Program** or **[Ctrl]+[F5]** to program the device.

Figure 3-6. Program Device from PSoC Creator



The onboard programmer supports only the RESET programming mode. When using the onboard programmer, the board can either be powered by the USB (VBUS) or by an external source such as an Arduino shield. If the board is already powered from another source, plugging in the USB programmer does not damage the board.



3.2.2 Using CY8CKIT-002 MiniProg3 Programmer and Debugger

The PSoC 4 on the Pioneer Kit can also be programmed using a MiniProg3 (CY8CKIT-002). To use MiniProg3 for programming, use the J6 connector on the board, as shown in Figure 3-7. With MiniProg3, programming is similar to the onboard programmer; however, the setup enumerates as a MiniProg3. Only the RESET programming mode is available.

The board can also be powered from the MiniProg3. To do this, select **Tool > Options**. In the Options window, expand **Program and Debug > Port Configuration**; click **MiniProg3** and select the settings shown in Figure 3-8. Click **Debug > Program** to program and power the board.

Note: The CY8CKIT-002 MiniProg3 is not part of the PSoC 4 Pioneer Kit contents. It can be purchased from the Cypress Online Store.

Figure 3-7. PSoC 4 Programming/Debug Using MiniProg3



Figure 3-8. MiniProg3 Configuration

Options		? ×
Project Management Text Editor Design Entry Language Support Program/Debug General Fonts and Colors Device Recognition Pot Configuration MiniProg3 FX2LP-SWD TrueTouchBridge DVKProg1 KitProg Environment	Active Processon	otocol: SWD
Restore All Defaults	OK Apply	Cancel

Note: See the Programmer User Guide for more information on programming using a MiniProg3.

3.3 USB-UART Bridge

The onboard PSoC 5LP can also act as a USB-UART bridge to transfer and receive data from the PSoC 4 device to the PC via the COM terminal software. When the USB mini-B cable is connected to J10 of the PSoC 4 Pioneer Kit, a device named **KitProg USBUART** is available under Ports (COM & LPT) in the device manager. For more details about the USB-UART functionality, see Using PSoC 5LP as USB-UART Bridge on page 66.

To use the USB-UART functionality in the COM terminal software, select the corresponding COM port as the communication port for transferring data to and from the COM terminal software.

The UART lines from PSoC 5LP are brought to the P12[6] (J8_9) and P12[7] (J8_10) pins of header J8. This interface can be used to send or receive data from any PSoC 4 design that has a UART by connecting the pins on header J8 to the RX and TX pins assigned in PSoC 4. The UART can be used as an additional interface to debug designs. This bridge can also be used to interface with other external UART-based devices. Figure 3-9 shows the connection between the RX and TX lines of the PSoC 5LP and PSoC 4. In this example, the PSoC 4 UART has been routed to the J3 header; the user must connect the wires between the PSoC 5LP RX and TX lines available on header J8.





Figure 3-9. Example RX and TX Line Connection of PSoC 5LP and PSoC 4

Table 3-2 lists the specifications supported by the USB-UART bridge.

Parameter	Supported Values
Baud Rate	1200, 2400, 4800, 9600, 19200, 38400, 57600, and 115200
Data Bits	8
Parity	None
Stop Bits	1
Flow Control	None
File transfer protocols supported	Xmodem, 1K Xmodem, Ymodem, Kermit, and Zmodem (only speeds greater than 2400 baud).

Table 3-2. Specifications Supported by USB-UART Bridge



3.4 USB-I2C Bridge

The PSoC 5LP also functions as a USB-I2C bridge. The PSoC 4 communicates with the PSoC 5LP using an I2C interface and the PSoC 5LP transfers the data over the USB to the USB-I2C software utility on the PC, called the Bridge Control Panel (BCP).

The BCP is available as part of the PSoC Programmer installation. This software can be used to send and receive USB-I2C data from the PSoC 5LP. When the USB mini-B cable is connected to header J10 on the Pioneer Kit, the **KitProg USB-I2C** is available under **Connected I2C/SPI/RX8 Ports** in the BCP.

Figure 3-10. Bridge Control Panel

Bridge Control Panel			
File Editor Chart Execute Iools Help 26 1			
			•
Select Port in the PortList, then try to Dpening Port Successfully Connected to KitProg/1222172 KitProg Version 2.00 Select Port in the PortList, then try to	2E03242400	Connect Button	*
<			Þ
(2) Reset Send	Connected I2C/SPI/RX8 Ports: KiIProg/1222172E03242400 COM3 COM6 COM6 COM6	► Power 5.50V → 33V → 25V → 1.8V	Protocol I2C SPI RX8 (UART)
1:1 Syntax:OK	Voltage: -		

To use the USB_I2C functionality, select the **KitProg USB-I2C** in the BCP. On successful connection, the **Connected** and **Powered** tabs turn green.



Bridge Control Panel	•	_ 🗆 💌
File Editor Chart Execute Tools Help		
Editor Chart Table File		
		~
		4
KitProg Version 2.00 Select Port in the PortList, then try to	connect	^
Opening Port		
Successfully Connected to KitProg/122217	2E03242400	
KitProg Version 2.00		Ξ
4		*
	Connected I2C/SPI/RX8 Ports:	
🔇 Reset 🎉: List 🐺 Send Send all strings: 📃	KitProg/1222172E03242400	I2C
Stop Repeat To file Repeat count 0	COM7 = () +3.3V ((SPI
Scan period, ms: 0	COM6 COM9	RX8 (UART)
1:1 Syntax:OK Connected	Powered Voltage: 5025 mV	
1:1 Syntax:OK Connected	rowered vollage: 5025 mv	

Figure 3-11. KitProg USB-I2C Connected in Bridge Control Panel

USB-I2C is implemented using the USB and I2C components of PSoC 5LP. The SCL (P12_0) and SDA (P12_1) lines from the PSoC 5LP are connected to SCL (P3_0) and SDA (P3_1) lines of the PSoC 4 I2C. The USB-I2C bridge currently supports I2C speed of 50 kHz, 100 kHz, 400 kHz, and 1 MHz.

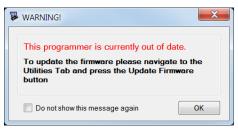
Refer to Using PSoC 5LP as USB-I2C Bridge on page 79 for building a project, which uses USB-I2C Bridge functionality.

3.5 Updating the Onboard Programmer Firmware

The firmware of the onboard programmer and debugger, PSoC 5LP, can be updated from PSoC Programmer. When a new firmware is available or when the KitProg firmware is corrupt (see Error in Firmware/Status Indication in Status LED on page 120), PSoC Programmer displays a warning indicating that new firmware is available.

Open PSoC Programmer from **Start > All Programs > Cypress > PSoC Programmer<version>**. When PSoC Programmer opens, a WARNING! window pops up saying that the programmer is currently out of date.

Figure 3-12.	Firmware	Update	Warning
--------------	----------	--------	---------



Click **OK** to close the window. On closing the warning window, the **Action and Results** window displays "Please navigate to the Utilities tab and click the Upgrade Firmware button".

Figure 3-13. Upgrade Firmware Message in PSoC Programmer

PSoC Programmer		
File View Options Help		
🖆 · 🗼 💿 BB 🕗		
Port Selection IProgr	ammer Utilities JTAG	
KitProg/0311172E03242400	Upgrade Firmware Click to upgrade connected device's firmware Erase Block Click to erase user specific flash block	
Device Family CY8C5rootLP		
Device		
CY8C5868LTI-LP039 *		
Actions	Results	
	Please navigate to the Utilities tab and click the Upgrade Firmwa	are button
Port Opened with Warnings at 3:54:30 PM	KitProg version Expecting 2.03, but found 2.02.	
Opening Port at 3:54:29 PM		Update Firmware mess
Device set to		displayed in Actions an
CY8C5868LTI-LP039 at 3:54:29 PM	262144 FLASH bytes	Results window
Device Family set to CY8C5xxxLP at 3:54:29 PM		

Click the **Utilities** tab and click the **Upgrade Firmware** button. On successful upgrade, the **Action and Results** window displays the firmware update message with the KitProg version.

PSoC Programmer	
File View Options He	lp
🖆 · 🗼 🔘 🕒	
Port Selection	Programmer Utilities JTAG
KitProg/0311172E0324240C	Upgrade Firmware Click to upgrade connected device's firmware Erase Block Click to erase user specific flash block
Device Family	
CY8C5xoocLP v	
Device	
CY8C5868LTI-LP039 *	
Actions	Results
	KitProg Version 2.03 KitProg Firmware version
Firmware Update Fini at 3:58:13 PM	
	Succeeded Firmware Update message
	Verifying Upgrading
	Initializing
Firmware Upgrade Sta	irted
at 3:58:01 PM	
Firmware Upgrade	

Figure 3-14. Firmware Updated in PSoC Programmer

4. Hardware

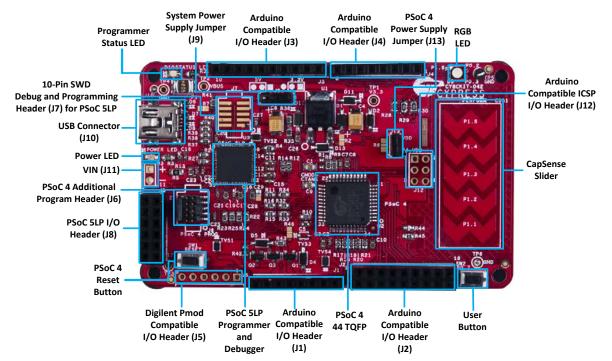


4.1 Board Details

The PSoC 4 Pioneer Kit consists of the following blocks:

- PSoC 4
- PSoC 5LP
- Power supply system
- Programming interfaces (J6, J7 unpopulated, J10)
- Arduino compatible headers (J1, J2, J3, J4, and J12 unpopulated)
- Digilent Pmod compatible header (J5 unpopulated)
- PSoC 5LP GPIO header (J8)
- CapSense slider
- Pioneer board LEDs
- Push buttons (Reset and User buttons)

Figure 4-1. PSoC 4 Pioneer Kit Details





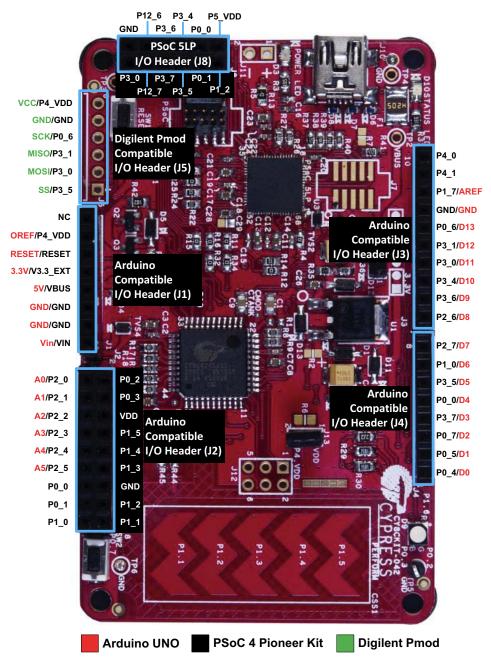


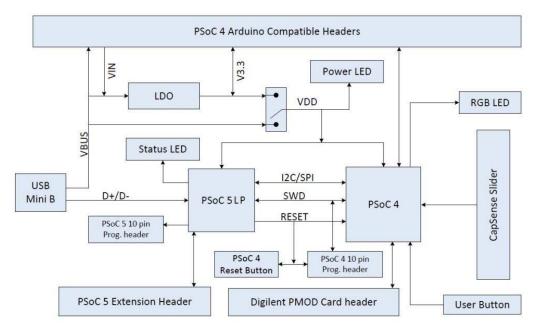
Figure 4-2. PSoC 4 Pioneer Kit Pin Mapping



4.2 Theory of Operation

This section provides the block-level description of the PSoC 4 Pioneer Kit.





The PSoC 4 is a new generation of programmable system-on-chip devices from Cypress for embedded applications. It combines programmable analog, programmable digital logic, programmable I/O, and a high-performance ARM Cortex-M0 subsystem. With the PSoC 4, you can create the combination of peripherals required to meet the application specifications.

The PSoC 4 Pioneer Kit features an onboard PSoC 5LP, which communicates through the USB to program and debug the PSoC 4 using serial wire debug (SWD). The PSoC 5LP also functions as a USB-I2C bridge and USB-UART bridge.

The Pioneer Kit has an RGB LED, a status LED, and a power LED. The RGB LED is connected to the PSoC 4 and the status LED is connected to the PSoC 5LP. For more information on the status LED, see section A.5 Error in Firmware/Status Indication in Status LED on page 120. This kit also includes a reset button that connects to the PSoC 4 XRES, a user button, and a five-segment CapSense slider, which can be used to develop touch-based applications. The PSoC 4 pins are brought out onto headers J1 to J4 on the kit to support Arduino shields. The PSoC 5LP pins are brought out onto header J8 to enable using the onboard PSoC 5LP to develop custom applications.

The PSoC 4 Pioneer Kit can be powered from the USB Mini B, the Arduino compatible header, or an external power supply. The input voltage is regulated by a low drop-out (LDO) regulator to 3.3 V. You can select between VBUS (5 V) and 3.3 V by suitably plugging the jumper onto the voltage selection header VDD.



4.3 Functional Description

4.3.1 PSoC 4

This kit uses the PSoC 4200 family device. PSoC 4200 devices are a combination of a microcontroller with programmable logic, high-performance analog-to-digital conversion, two opamps with comparator mode, and commonly used fixed-function peripherals. For more information, refer to the PSoC 4 web page and the PSoC 4200 family datasheet.

Features

- 32-bit MCU subsystem
 - □ 48 MHz ARM Cortex-M0 CPU with single cycle multiply
 - □ Up to 32 KB of flash with read accelerator
 - □ Up to 4 KB of SRAM
- Programmable analog
 - Two opamps with reconfigurable high-drive external and high-bandwidth internal drive, comparator modes, and ADC input buffering capability
 - 12-bit 1-Msps SAR ADC with differential and single-ended modes; channel sequencer with signal averaging
 - □ Two current DACs (IDACs) for general-purpose or capacitive sensing applications on any pin
 - $\hfill\square$ Two low-power comparators that operate in deep sleep
- Programmable digital
 - Four programmable logic blocks called universal digital blocks (UDBs), each with eight Macrocells and data path
 - Cypress-provided peripheral component library, user-defined state machines, and Verilog input
- Low power 1.71 to 5.5 V operation
 - □ 20-nA Stop mode with GPIO pin wakeup
 - □ Hibernate and Deep-Sleep modes allow wakeup-time versus power trade-offs
- Capacitive sensing
 - Cypress Capacitive Sigma-Delta (CSD) provides best-in-class SNR (greater than 5:1) and water tolerance
 - Cypress-supplied software component makes capacitive sensing design easy
 - □ Automatic hardware tuning (SmartSense[™])
- Segment LCD drive
 - □ LCD drive supported on all pins (common or segment)
 - Operates in Deep-Sleep mode with 4 bits per pin memory
- Serial communication
 - Two independent run-time reconfigurable serial communication blocks (SCBs) with re-configurable I2C, SPI, or UART functionality
- Timing and pulse-width modulation
 - □ Four 16-bit Timer/Counter Pulse-Width Modulator (TCPWM) blocks
 - □ Center-aligned, Edge, and Pseudo-random modes
 - Comparator-based triggering of Kill signals for motor drive and other high-reliability digital logic applications
- Up to 36 programmable GPIOs
 - □ 44-pin TQFP, 40-pin QFN, and 28-pin SSOP packages
 - □ Any GPIO pin can be CapSense, LCD, analog, or digital
 - Drive modes, strengths, and slew rates are programmable
- PSoC Creator design environment
 - Integrated development environment (IDE) provides schematic design entry and build (with analog and digital automatic routing)



- Applications Programming Interface (API) component for all fixed-function and programmable peripherals
- Industry-standard tool compatibility
 - After schematic entry, development can be done with ARM-based industry-standard development tools

For more information see the CY8C42 family datasheet.

4.3.2 PSoC 5LP

An onboard PSoC 5LP is used to program and debug PSoC 4. The PSoC 5LP connects to the USB port of the PC through a USB Mini B connector and to the SWD interface of the PSoC 4 device.

PSoC 5LP is a true system-level solution providing MCU, memory, analog, and digital peripheral functions in a single chip. The CY8C58LPxx family offers a modern method of signal acquisition, signal processing, and control with high accuracy, high bandwidth, and high flexibility. Analog capability spans the range from thermocouples (near DC voltages) to ultrasonic signals. For more information, refer to the PSoC 5LP web page.

Features

- 32-bit ARM Cortex-M3 CPU core
 - DC to 67-MHz operation
 - Flash program memory, up to 256 KB, 100,000 write cycles, 20-year retention, and multiple security features
 - Up to 32-KB flash error correcting code (ECC) or configuration storage
 - □ Up to 64 KB SRAM
 - 2-KB electrically erasable programmable read-only memory (EEPROM) memory, 1 M cycles, and 20 years retention
 - 24-channel direct memory access (DMA) with multilayer AHB bus access a.Programmable chained descriptors and priorities
 b.High bandwidth 32-bit transfer support
- Low voltage, ultra low power
 - □ Wide operating voltage range: 0.5 V to 5.5 V
 - □ High-efficiency boost regulator from 0.5 V input to 1.8 V to 5.0 V output
 - □ 3.1 mA at 6 MHz
 - Low power modes including: a.2-µA sleep mode with real time clock (RTC) and low-voltage detect (LVD) interrupt b.300-nA hibernate mode with RAM retention
- Versatile I/O system
 - □ 28 to 72 I/Os (62 GPIOs, 8 SIOs, 2 USBIOs)
 - □ Any GPIO to any digital or analog peripheral routability
 - □ LCD direct drive from any GPIO, up to 46×16 segments
 - □ CapSense support from any GPIO[3]
 - □ 1.2 V to 5.5 V I/O interface voltages, up to 4 domains
 - □ Maskable, independent IRQ on any pin or port
 - □ Schmitt-trigger transistor-transistor logic (TTL) inputs
 - □ All GPIOs configurable as open drain high/low, pull-up/pull-down, High-Z, or strong output
 - □ Configurable GPIO pin state at power-on reset (POR)
 - □ 25 mA sink on SIO
- Digital peripherals
 - □ 20 to 24 programmable logic device (PLD) based universal digital blocks (UDBs)
 - □ Full CAN 2.0b 16 RX, 8 TX buffers
 - □ Full-Speed (FS) USB 2.0 12 Mbps using internal oscillator



- □ Four 16-bit configurable timers, counters, and PWM blocks
- 67-MHz, 24-bit fixed point digital filter block (DFB) to implement finite impulse response (FIR) and infinite impulse response (IIR) filters
- Library of standard peripherals
 - a.8-, 16-, 24-, and 32-bit timers, counters, and PWMs

b.Serial peripheral interface (SPI), universal asynchronous transmitter receiver (UART), and I2C

c.Many others available in catalog

Library of advanced peripherals

a.Cyclic redundancy check (CRC)

- b.Pseudo random sequence (PRS) generator
- c.Local interconnect network (LIN) bus 2.0
- d.Quadrature decoder
- □ Analog peripherals (1.71 V ≤ VDDA ≤ 5.5 V)
- □ 1.024 V ±0.1% internal voltage reference across –40 °C to +85 °C
- □ Configurable delta-sigma ADC with 8- to 20-bit resolution
- □ Sample rates up to 192 ksps
- □ Programmable gain stage: ×0.25 to ×16
- □ 12-bit mode, 192 ksps, 66-dB signal to noise and distortion ratio (SINAD), ±1-bit INL/DNL
- □ 16-bit mode, 48 ksps, 84-dB SINAD, ±2-bit INL, ±1-bit DNL
- □ Up to two SAR ADCs, each 12-bit at 1 Msps
- □ Four 8-bit 8 Msps current IDACs or 1-Msps voltage VDACs
- □ Four comparators with 95-ns response time
- □ Four uncommitted opamps with 25-mA drive capability
- Four configurable multifunction analog blocks. Example configurations are programmable gain amplifier (PGA), transimpedance amplifier (TIA), mixer, and sample and hold
- □ CapSense support
- Programming, debug, and trace
 - □ JTAG (4 wire), SWD (2 wire), single wire viewer (SWV), and TRACEPORT interfaces
 - □ Cortex-M3 flash patch and breakpoint (FPB) block
 - □ Cortex-M3 Embedded Trace Macrocell[™] (ETM[™]) generates an instruction trace stream
 - □ Cortex-M3 data watchpoint and trace (DWT) generates data trace information
 - □ Cortex-M3 Instrumentation Trace Macrocell (ITM) can be used for printf-style debugging
 - DWT, ETM, and ITM blocks communicate with off-chip debug and trace systems via the SWV or TRACEPORT
 - **D** Bootloader programming supportable through I2C, SPI, UART, USB, and other interfaces
- Precision, programmable clocking
 - □ 3- to 62-MHz internal oscillator over full temperature and voltage range
 - □ 4- to 25-MHz crystal oscillator for crystal PPM accuracy
 - □ Internal PLL clock generation up to 67 MHz
 - □ 32.768-kHz watch crystal oscillator
 - □ Low-power internal oscillator at 1, 33, and 100 kHz

For more, see the CY8C58LPxx family datasheet.



4.3.3 Power Supply System

The power supply system on this board is versatile, allowing the input supply to come from the following sources:

- 5-V power from onboard USB programming header J10
- 5-V to 12-V power from Arduino shield using J1_01 header
- VTARG power from the onboard SWD programming using J6 or J7
- VIN J11

The PSoC 4 and PSoC 5LP are powered with either a 3.3 V or 5 V source. The selection between 3.3 V and 5 V is made through the J9 jumper. The board can supply 3.3 V and 5 V to the I/O headers and receive 3.3 V from the I/O headers. The board can also be powered with an external power supply through the VIN (J11) header; the allowed voltage range for the VIN is 5 V to 12 V. The LDO regulator regulates the VIN down to 3.3 V. Figure 4-4 shows the power supply block diagram and protection circuitry.

Note: The 5-V domain is directly powered by the USB (VBUS). For this reason, this domain is unregulated.

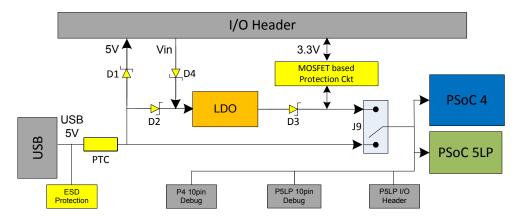
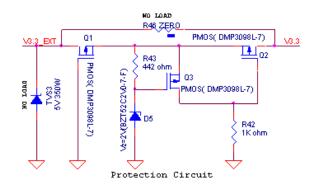


Figure 4-4. Power Supply Block Diagram with Protection Circuits





4.3.3.1 Protection Circuit

The power supply rail has reverse-voltage, over-voltage, short circuits, and excess current protection features, as seen in Figure 4-4.

- The Schottky diode (D1) ensures power cannot be supplied to the 5-V domain of the board from the I/O header.
- The series protection diode (D2) ensures VIN (power supply from the I/O header) does not back power the USB.
- The Schottky diode (D3) ensures 3.3 V from I/O header does not back power the LDO.
- The series protection diode (D4) ensures that the reverse-voltage cannot be supplied from the VIN to the regulator input.
- A PTC resettable fuse is connected to protect the computer's USB ports from shorts and overcurrent.
- The MOSFET-based protection circuit provides over-voltage and reverse-voltage protection to the 3.3-V rail. The PMOS Q1 protects the board components from a reverse-voltage condition. The PMOS Q2 protects the PSoC from an over-voltage condition. The PMOS Q2 will turn off when a voltage greater than 4.2 V is applied, protecting the PSoC 4.
- The output voltage of the LDO is adjusted such that it takes into account the voltage drop across the Schottky diode and provides 3.3 V.

4.3.3.2 Procedure to Measure PSoC 4 Current Consumption

The following three methods are supported for measuring current consumption of the PSoC 4 device.

When the board is powered through the USB port (J10), remove jumper J13 and connect an ammeter, as shown in Figure 4-5.

Figure 4-5. PSoC 4 Current Measurement when Powered from USB Port



- When using a separate power supply for the PSoC 4 with USB powering (regulator output on the USB supply must be within 0.5 V of the separate power supply).
 - Remove jumper J13. Connect the positive terminal of voltage supply to the positive terminal of the ammeter and the negative terminal of the ammeter to the lower pin of J13. Figure 4-6 shows the required connections.





Figure 4-6. PSoC 4 Current Measurement when Powered Separately

- When the PSoC 4 is powered separately and the PSoC 5LP is not powered, make these changes to avoid leakage while measuring current:
 - Remove the zero-ohm resistors R24 and R25. Removing these resistors will affect the USB-I2C functionality.
 - Remove R11, R15, and R16, which are meant for programming the PSoC 4. Removing these resistors disables the PSoC 5LP capability for programming.
 - □ Connect an ammeter between pins 1 and 2 of header J13 to measure current.

Figure 4-7. Zero-ohm Resistor Position



R25, R24 R32, R33, R34

4.3.4 Programming Interface

The kit allows programming and debugging of the PSoC 4 in two modes:

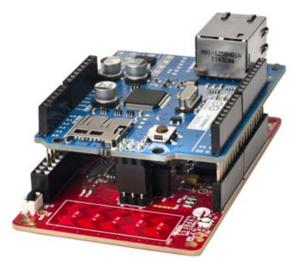
- Using the Onboard PSoC 5LP Programmer and Debugger
- Using CY8CKIT-002 MiniProg3 Programmer and Debugger



4.3.5 Arduino Compatible Headers (J1, J2, J3, J4, and J12 - unpopulated)

This kit has five Arduino compatible headers; J1, J2, J3, J4 and J12. You can develop applications based on the Arduino shield's hardware.

Figure 4-8. Arduino Header

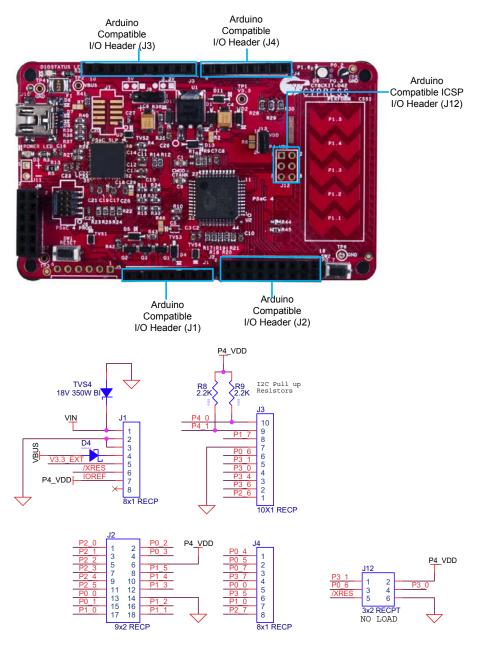


The J1 header contains I/O pins for reset, internal reference voltage (IOREF), and power supply line. The J2 header is an analog port. It contains I/O pins for SAR ADC, comparator, and opamp. The J3 header is primarily a digital port. It contains I/O pins for PWM, I2C, SPI, and analog reference. The J4 header is also a digital port. It contains I/O pins for UART and PWM. The J12 header is an Arduino ICSP compatible header for the SPI interface. This header is not populated. Refer to the "No Load Components" section of A.6 Bill of Materials (BOM) on page 121 for the header part number.

Note: The PSoC 4 pin P0[0] is connected to both pin 13 of the J2 header and pin 5 of the J4 header. Similarly, the PSoC 4 pin P1[0] is connected to both pin 17 of the J2 header and pin 7 of the J4 header. Therefore, when using P0[0] or P1[0] from either the J2 or J4 header, there should not be any external signal connected to the other header.



Figure 4-9. Arduino Compatible Headers



(J1-J4) Arduino Compatible Headers

4.3.5.1 Additional Functionality of Header J2

The J2 header is a 9×2 header that supports Arduino shields. The port 0, port 1, and port 2 pins of PSoC 4 are brought to this header. The port 1 pins additionally connect to the onboard CapSense slider through 560- Ω resistors. When the CapSense feature is not used, remove these resistors to ensure a better performance with these pins.



4.3.5.2 Functionality of Unpopulated Header J12

The J12 header is a 2×3 header that supports Arduino shields. This header is used on a small subset of shields and is unpopulated on the PSoC 4 Pioneer Kit. Note that the J12 header only functions in 5.0 V mode. To ensure proper shield functionality, ensure the power jumper is connected in 5.0 V mode.

4.3.6 Digilent Pmod Compatible Header (J5 - unpopulated)

This port supports Digilent Pmod peripheral modules. Pmods are small I/O interfaces, which connect with the embedded control boards through either 6- or 12-pin connectors. The PSoC Pioneer Kit supports the 6-pin Pmod type 2 (SPI) interface. For Digilent Pmod cards, go to www.digilentinc.com.

This header is not populated on the PSoC 4 Pioneer Kit. You must populate this header before connecting the Pmod daughter cards. Refer to the "No Load Components" section of A.6 Bill of Materials (BOM) on page 121 for the header part number.

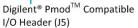
Figure 4-10. Pmod Connection

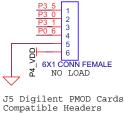




Figure 4-11. Digilent Pmod Interface







See A.2 Pin Assignment Table on page 116 for details on the pin descriptions for the J5 header.

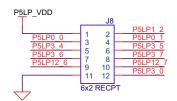
4.3.7 PSoC 5LP GPIO Header (J8)

A limited set of PSoC 5LP pins are brought to this header. Refer to 6.3 Developing Applications for PSoC 5LP on page 88 for details on how to develop custom applications. See A.2 Pin Assignment Table on page 116 for pin details.

Figure 4-12. PSoC 5LP GPIO Header (J8)



PSoC 5LP I/O Header (J8)



PSoC 5LP GPIO Extension Header



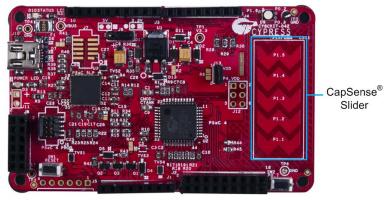
4.3.8 CapSense Slider

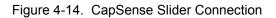
The kit has a five-segment linear capacitive touch slider on the board, which is connected to pins P1[1] to P1[5] of the PSoC 4 device.

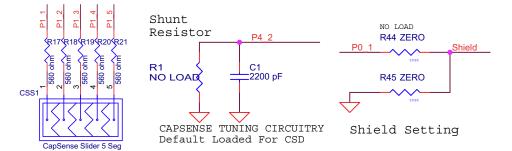
The modulation capacitor (Cmod) is connected to pin P4[2] and an optional bleeder resistor (R1) can be connected across the Cmod. This board supports CapSense designs that enable waterproofing.

The waterproofing design uses a concept called shield, which is a conductor placed around the sensors. This shield must be connected to a designated shield pin on the device to function. The shield must be connected to the ground when not used. On the PSoC 4 Pioneer Kit, the connection of the shield to the pin or to the ground is made by resistors R44 and R45, respectively. By default, R45 is mounted on the board, which connects the shield to the ground. Populate R44 when evaluating waterproofing designs, which will connect the shield to the designated pin, P0[1]. This shield is different from the Arduino shields, which are boards that connect over the Arduino header. Refer to the CapSense Design Guide for further details related to CapSense.











4.3.9 Pioneer Board LEDs

The PSoC 4 Pioneer board has three LEDs. A green LED (D10) indicates the status of the programmer. See A.5 Error in Firmware/Status Indication in Status LED for a detailed list of LED indications. An amber LED (D3) indicates status of power supplied to the board. The kit also has a general-purpose tricolor LED (D9) for user applications that connect to specific PSoC 4 pins.

Figure 4-15 shows the indication of all these LEDs on the board. Figure 4-16 and Figure 4-17 detail the LED schematic.

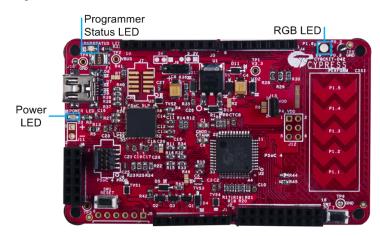
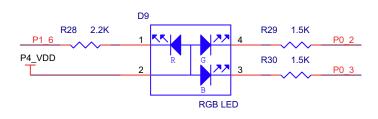


Figure 4-15. Pioneer Kit LEDs

Figure 4-16. Status LED and Power LED



Figure 4-17. RGB LED



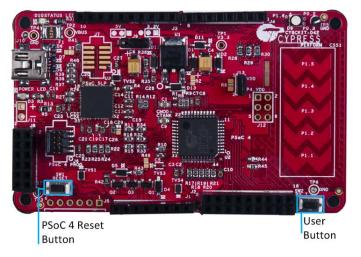


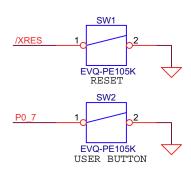
4.3.10 Push Buttons

The kit contains a Reset push button and a User push button, as shown in Figure 4-18.

The Reset button is connected to the XRES pin of PSoC 4 and is used to reset the onboard PSoC 4 device. The User button is connected to P0[7] of PSoC 4 device. Both the push buttons connect to ground on activation (active low).

Figure 4-18. Push Buttons





5. Code Examples

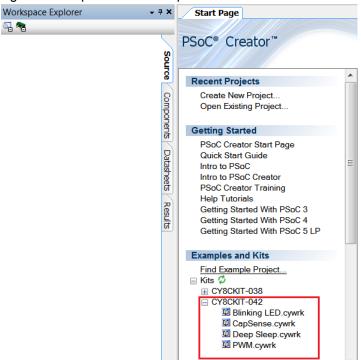


The code examples described in this chapter introduce the functionality of the PSoC 4 device and the onboard components. To access the examples, download the CD ISO image or setup files from the kit web page. The code examples will be available in the firmware folder in the install location. For a list of all code examples available with PSoC Creator visit PSoC 3/4/5 Code Examples webpage.

Follow these steps to open and program code examples:

- 1. Launch PSoC Creator from the Start menu.
- Open the code example by clicking <Project.cywrk> below Examples and Kits > Find Example Project > Kits > CY8CKIT-042.

Figure 5-1. Open Code Example from PSoC Creator





3. Build the code example by clicking Build > Build <Project name> to generate the hex file.

Figure 5-2. Build Project from PSoC Creator File Edit View Project Build Debug Tools Window Help Build Blinking LED Shift+F6 🛅 🎦 💣 🗃 🗐 🖉 🕰 🕰 🖽 - 🚵 🕸 💕 👹 🖕 🖕 Clean Blinking LED licrosoft Sa ×. Clean and Build Blinking LED Workspace Explorer tart Page ū, 🥿 Cancel Build Ctrl+Break Workspace 'Blinking LED' (1 P **Compile File** 🖻 😼 *Project 'Blinking LED' TopDesign.cysch Generate Application R\$ 🖞 Blinking LED.cydwr 🗄 🧰 Header Files 1 Generate Project Datasheet b device.h Source Files 📄 main c

- To program, connect the board to a computer using the USB cable connected to port J10, as described in section 3.2 Programming and Debugging PSoC 4 on page 23. The board is detected as KitProg.
- 5. Click **Debug > Program** from PSoC Creator.

Figure 5-3. Program Device from PSoC Creator <u>File Edit View Project Build Debug Tools Window Help</u> 🔓 🖞 🧟 😂 🖌 🕼 🖓 🖓 🖓 Windows - 🔍 🔍 ۲ Ctrl+F5 🖽 • 🚵 📽 👹 🌺 🖕 沙 😘 Program - 10 Select Debug Target... Workspace Explorer (1 project) 澎 sign.cysch م 🗣 × Debug F5 Workspace 'Blinking LED' (1 Projects) 斑 Debug without Programming Alt+F5 🖻 🔁 *Project 'Blinking LED' [CY8C4 TopDesign.cysch Attach to Running Target... Blinking LED.cydwr ø Toggle Breakpoint F9 🗄 ն Header Files b device.h New Breakpoint 🗄 🗀 Source Files --- ci main.c 50 Delete All Breakpoints Ctrl+Shift+F9 🗄 🧀 Generated Source Enable All Breakpoints 🗄 🧀 PSoC4 🖻 🧰 Clock D

6. If the device is not yet acquired, PSoC Creator will open the programming window. Select **KitProg/** and click the **Port Acquire** button.

Select Debug Target

 KitProg/1222172E03242400
 KitProg/1222172E03242400

 POWER = 3
 VOLTAGE_ADC = 3316

 FREQUENCY = 2000000
 PROTOCOL = SWD

 KitProg version Expecting 2.00, but found 1.03
 KitProg version Expecting 2.00, but found 1.03

 Show all targets
 Port Setting
 Port Acquire

 OK
 OK
 OK

Figure 5-4. Acquire Device from PSoC Creator



7. After the device is acquired, it is shown in a tree structure below the **KitProg**. Now, click the **Connect** button.

Figure 5-5. Connect Device from PSoC Creator

Select Debug Target	
E-∑ KitProg/1222172E03242400 PSoC 4 CY8C4245AXI-483	PSoC 4 CY8C4245AXI-483 PSoC 4 (ARM CM0) Silicon ID: 0x0BB11477 Cypress ID: 0x04C81193 Revision: PRODUCTION Target unacquired
Show all targets	Connect
	ОК

8. Click **OK** to exit the window and start programming.

Figure 5-6. Program Device from PSoC Creator

Select Debug Target	? ×
E-5 KitProg/1222172E03242400 └∽ PSoC 4 CY8C4245AXI-483 (Connected)	PSoC 4 CY8C4245AXI-483 PSoC 4 (ARM CM0) Silicon ID: 0x0BB11477 Cypress ID: 0x04C81193 Revision: PRODUCTION Target acquired
Show all targets	<u>D</u> isconnect
	ОК



5.1 **Project: Blinking LED**

5.1.1 Project Description

This example uses a pulse-width modulator (PWM) to illuminate the RGB LED. The PWM output is connected to pin P0_3 (blue) of the RGB LED. The frequency of blinking is set to 1 Hz with a duty cycle of 50 percent. The blinking frequency and duty cycle can be varied by varying the period and compare value respectively.

Note: The PSoC 4 Pioneer Kit is factory-programmed with this example.

P4_VDD Vdd Blue 1.5K Resistor PWM **PWM** ov ЬÐ un 🖃 ā count 1 B start cc -1 Pin BlueLED line line_n 🖃 Clock Sclock 10 kHz interrupt F

Figure 5-7. PSoC Creator Schematic Design of Blinking LED Project

5.1.2 Hardware Connections

No specific hardware connections are required for this project because all connections are hardwired on the board. Open *Blinking LED.cydwr* in the Workspace Explorer and select the suitable pin.

Table 5-1. Pin Connection

Pin Name	Port Name
Pin_BlueLED	P0_3 (Blue)

Figure 5-8. Pin Selection for Blinking LED Project

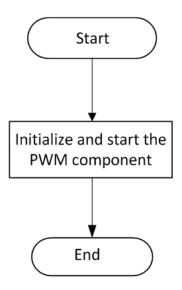
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~	Alias	Name 🗸	Port		Lock
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~		Pin_BlueLED	P0[3] COMP2:inn	27 💌	<b>V</b>



5.1.3 Flow Chart

Figure 5-9 shows the flow chart of code implemented in *main.c.*

Figure 5-9. Blinking LED Project Flow Chart



5.1.4 Verify Output

Build and program the code example onto the device. Observe the frequency and duty cycle of the blinking LED. Change the period and compare value in the PWM component, as shown in Figure 5-10. Rebuild and reprogram the device to vary the frequency and duty cycle.

Figure 5-10. PWM Component Configuration Window

Configure 'TCPWM_P4'								9	X
Name: PWM									
Configuration PWM	Built-in								۹ ۵
Prescaler:	1x ·	•	Input	Present	Mode				^
PWM align:	Left align	•	reload		Rising	edge		-	
PWM mode:	PWM		start	✓	Level Rising			•	
Dead time cycle:	0		stop switch		Rising	<u> </u>		•	
Stop signal event:		 ▼ 	count	v	Level			-	
		• _							=
Kill signal event:	Asynchronous	▼		Regis	ter	Swap	Register	Buf	
Output line signal:	Direct output	•	Period	10000			65535		
Output line_n signal:	Direct output	▼	Compare	1000			65535		
Interrupt									
On terminal count									
On compare/capture	e count								
	F	PWM, left aligne	be						
		www.ieit angri	Ju .						
10000			/					_	T
Datasheet				ОК		Apply		Canc	el

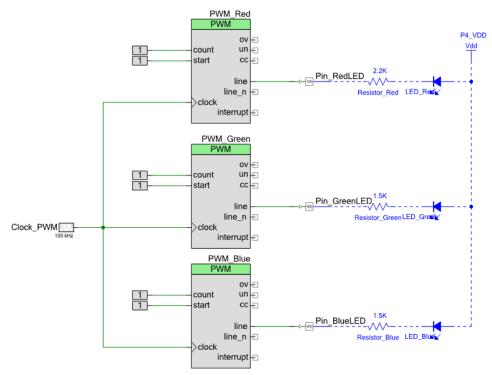


5.2 Project: PWM

5.2.1 Project Description

This code example demonstrates the use of the PWM component. The project uses three PWM outputs to set the color of RGB LED on the Pioneer Kit. The LED cycles through seven colors – violet > indigo > blue > green > yellow > orange > red (VIBGYOR). Each color is maintained for a duration of one second. The different colors are achieved by changing the pulse width of the PWMs.

Figure 5-11. PSoC Creator Schematic Design of PWM Project



5.2.2 Hardware Connections

No specific hardware connections are required for this project because all connections are hardwired on the board. Open *PWM.cydwr* in the Workspace Explorer and select the suitable pins.

Table 5-2. Pin Connections

Pin Name	Port Name
Pin_RedLED	P1_6 (Red)
Pin_GreenLED	P0_2 (Green)
Pin_BlueLED	P0_3 (Blue)

Figure 5-12. Pin Selection for PWM Project

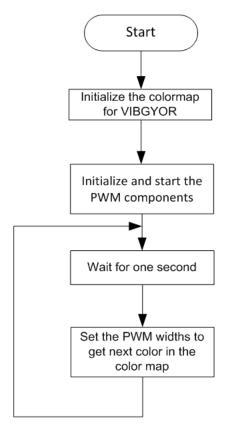
Alias	Name 🗠	Port		Lock
	Pin_BlueLED	P0[3] COMP2:inn	27 🔻	
	Pin_GreenLED	P0[2] COMP2:inp, SCB0:spi_ssel[3]	26 🔻	
	Pin RedLED	P1[6] OA0:vplus alt	43 🔻	



5.2.3 Flow Chart

Figure 5-13 shows the flow chart of code implemented in main.c.

Figure 5-13. PWM Project Flow Chart



5.2.4 Verify Output

Build and program the code example, and reset the device. Observe the RGB LED cycles through the color pattern.

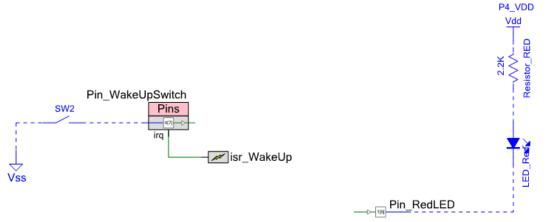


5.3 Project: Deep Sleep

5.3.1 Project Description

This project demonstrates the low-power functionality of the PSoC 4. The LED is turned on for one second to indicate Active mode; then, the device enters Deep-Sleep mode. When switch SW2 is pressed, the device wakes up and the LED is turned on for one second and then goes back into Deep-Sleep mode.

Figure 5-14. PSoC Creator Schematic Design of Deep-Sleep Project



5.3.2 Hardware Connections

No extra connections are required for the project functionality because the connections are hardwired onto the board. To make low-power measurements using this project, refer to the use case detailed in section 4.3.3.2 Procedure to Measure PSoC 4 Current Consumption on page 38.

Open *Deep Sleep.cydwr* in the Workspace Explorer and select the suitable pin.

Table 5-3. Pin Connection

Pin Name	Port Name		
Pin_RedLED	P1_6 (Red)		
Pin_WakeUpSwitch	P0_7		

Figure 5-15. Pin Selection for Deep-Sleep Project

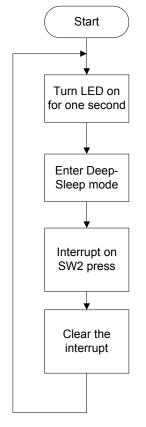
Alias	Name 🗠	Port			n	Lock
	Pin_RedLED	P1[6] OA0:vplus_alt	•	43	-	✓
	Pin_WakeUpSwitch	P0[7] SRSS:wakeup, SCB1:spi_ssel[0]	•	31	-	V



5.3.3 Flow Chart

Figure 5-16 shows the flow chart of code implemented in *main.c.*

Figure 5-16. Deep-Sleep Project Flow Chart



5.3.4 Verify Output

Build and program the code example, and reset the device. LED is on for one second and turns off, which indicates that the device has entered Deep-Sleep mode. Press SW2 switch to wake up the device from Deep-Sleep mode and enter Active mode. The device goes back to sleep after one second.

Note: When the device is in Deep-Sleep mode, the programmer must reacquire the device before programming can start.



5.4 Project: CapSense

This code example can be executed in two ways – with and without CapSense tuning. The same project can be used to demonstrate the CapSense functionality as well as CapSense tuning using the Tuner Helper GUI in PSoC Creator. This is done by commenting and uncommenting the line #define ENABLE_TUNER in the *main.c* file of the code example. PSoC Creator does not compile the code under the #ifdef (if defined) statement when the #define statement is commented (/ *..... */ or //). Similarly, when the #define statement is uncommented, the code required for working with Tuner GUI is compiled. By default, the project is set to work without CapSense tuning by commenting the #define.

5.4.1 CapSense (Without Tuning)

5.4.1.1 Project Description

This code example demonstrates CapSense on PSoC 4. The example uses the five-segment CapSense slider on the board. Each capacitive sensor on the slider is scanned using Cypress's CapSense Sigma Delta (CSD) algorithm implemented in the CapSense component. This project is pre-tuned to take care of the board parasitics. For more information on the CapSense component and CapSense tuning, see the CapSense component datasheet in PSoC Creator.

In this code example, the brightness of the green and red LEDs are varied, based on the position of the user's finger on the CapSense slider.

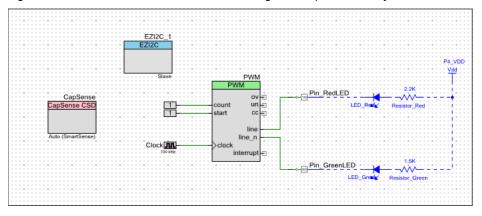


Figure 5-17. PSoC Creator Schematic Design of CapSense Project

Note: The EzI2C component is not used when tuning is disabled.



5.4.1.2 Hardware Connections

No specific hardware connections are required for this project because all connections are hardwired on the board. Open *CapSense.cydwr* in the Workspace Explorer and select the suitable pins.

Pin Name	Port Name
CapSense:Cmod	P4_2
CapSense:Sns[0]	P1_1
CapSense:Sns[1]	P1_2
CapSense:Sns[2]	P1_3
CapSense:Sns[3]	P1_4
CapSense:Sns[4]	P1_5
Pin_GreenLED	P0_2 (Green)
Pin_RedLED	P1_6 (Red)
EZI2C_1:scl	P3_0 (SCL)
EZI2C_1:sda	P3_1 (SDA)

Table 5-4. Pin Connection

Note: The I2C communication lines are not used when tuning is disabled.

Figure 5-18. Pin Selection for CapSense Project

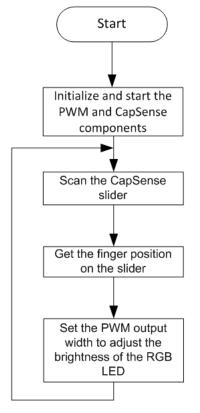
Alias	Name 🛛	Port		Port		Pi	in	Lock
Cmod	\CapSense:Cmod\	P4[2] CSD:c_mod, SCB0:spi_clk	•	22	•			
LinearSlider0_e0LS	\CapSense:Sns[0]\	P1[1] OA0:vminus, TCPWM2:line_out_compl	•	38	•	V		
LinearSlider0_e1LS	\CapSense:Sns[1]\	P1[2] OA0:vout_10x, TCPWM3:line_out	•	39	-	V		
LinearSlider0_e2LS	\CapSense:Sns[2]\	P1[3] OA1:vout_10x, TCPWM3:line_out_compl	•	40	•	V		
LinearSlider0_e3LS	\CapSense:Sns[3]\	P1[4] OA1:vminus	•	41	•	V		
LinearSlider0_e4LS	\CapSense:Sns[4]\	P1[5] OA1:vplus	•	42	•			
	\EZI2C_1:scl\	P3[0] TCPWM0:line_out, SCB1:uart_rx, SCB1:i2c scl, SCB1:spi mosi	•	11	-			
	\EZI2C_1:sda\	P3[1] TCPWM0:line_out_compl, SCB1:uart_tx, SCB1:i2c sda, SCB1:spi miso	•	12	-	V		
	Pin_GreenLED	<pre>P0[2] COMP2:inp, SCB0:spi_ssel[3]</pre>	•	26	•	V		
	Pin RedLED	P1[6] OA0:vplus alt	•	43	•	V		



5.4.1.3 Flow Chart

Figure 5-19 shows the flow chart of code implemented in *main.c.*

Figure 5-19. CapSense Project Flow Chart



5.4.1.4 Verify Output

The brightness of the green and red LEDs are varied based on the position of the user's finger on the CapSense slider. When the finger is on segment 5 (P1[5]) of the slider, the green LED is brighter than the red LED; when the finger is on segment 1 (P1[1]) of the slider, the red LED is brighter than the green LED.



5.4.2 CapSense (With Tuning)

5.4.2.1 Project Description

This code example demonstrates CapSense tuning on PSoC 4 using the "Tuner" to monitor CapSense outputs. The CapSense outputs such as rawcounts, baseline, and signal (difference count) can be monitored on the Tuner GUI. The project uses the auto-tuning feature, which sets all CapSense parameters to the optimum values automatically. The parameter settings can be monitored in the GUI but cannot be altered. In the manual tuning method, parameter settings can be changed in the GUI and the resulting output can be seen.

The code example uses the five-segment CapSense slider on the board. Each capacitive sensor on the slider is scanned using Cypress's CapSense Sigma Delta (CSD) algorithm implemented in the CapSense component. The code uses tuner APIs. The tuner API CapSense_TunerComm() is used in the main loop to scan sensors, which also sends the CapSense variables RawCounts, Baseline, and Difference Counts (Signal) to the PC GUI through I2C communication.

In this example, the brightness of the green and red LEDs are varied, based on the position of the user's finger on the CapSense slider.

See Figure 5-17 for the project schematic.

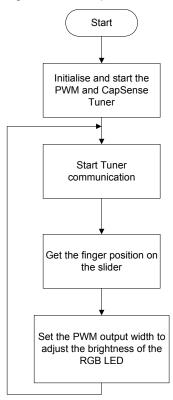
5.4.2.2 Hardware Connections

No specific hardware connections are required for this project because all connections are hardwired on the board. Open *CapSense.cydwr* in the Workspace Explorer and select the suitable pins.

See Table 5-4 and Figure 5-18 for the CapSense project pin connections.

5.4.2.3 Flow Chart

Figure 5-20. CapSense with Tuning Flow Chart





5.4.2.4 Launching Tuner GUI

The Tuner GUI from PSoC Creator should be up and running for the code example to work. To launch the GUI follow these steps:

1. Go to the project's *TopDesign.cysch* file.

Figure 5-21. Top Design File

Workspace Explorer (1 project)	→ ∓ X
a -	
🖾 Workspace 'CapSense' (1 Projects)	*
🖻 📴 Project 'CapSense' [CY8C4245AXI-483]	(v 🗌
- E TopDesign.cvsch	Source
- 🔐 CapSense.cydwr	8
🗄 🗁 Header Files	
h device.h	Component
🗄 🗁 Source Files	Do Do
i i main.c	= ents

2. To open the tuner, right-click on the CapSense_CSD component in PSoC Creator and click Launch Tuner.

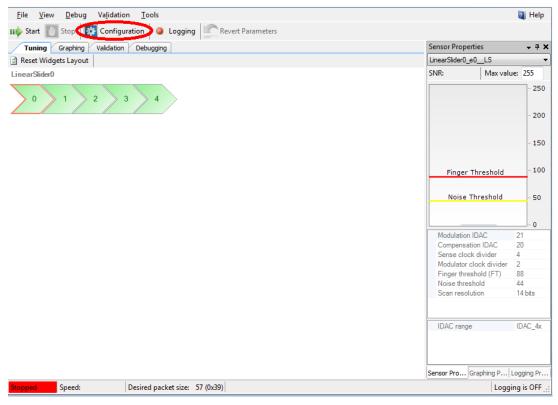
Figure 5-22. Launch Tuner

CapSer CapSenseC		
ANNO 250	Cu <u>t</u>	Ctrl+X
	<u>С</u> ору	Ctrl+C
EL	<u>P</u> aste	Ctrl+V
×	<u>D</u> elete	Del
	Select <u>A</u> ll	Ctrl+A
	Zoom	•
	Shape	•
	Configure	
project	Disab <u>l</u> e	1
jes co	Open Datasheet	t
	Find Example Pr	oject
	Open Compone	ent Web Page
	Launch Tuner	
	Generate Macro	,
	Show in analog	editor



3. The Tuner GUI opens. Click **Configuration** to open the configuration window.

Figure 5-23. Tuner GUI



4. Set the I2C communication parameters, as shown in the following figure.

Figure 5-24. I2C Communication

Tuner Communication Setup	? ×
Ports: KitProg/1508111801123400 Port Information KitProg Version 2.12	Port Configuration I2C address: 8 Sub-address: 2-Bytes I2C Speed 1 MHz 400 kHz 100 kHz 50 kHz
	OK Cancel

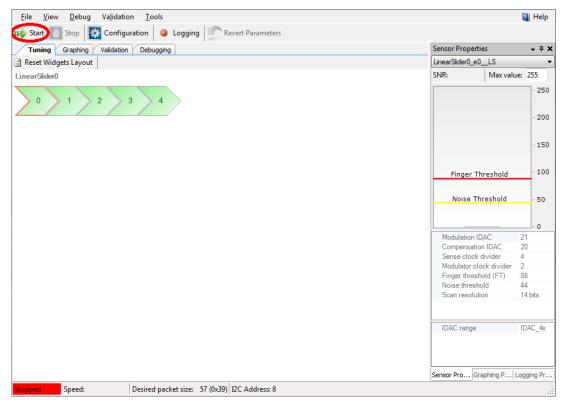
5. Click **OK** to apply the settings.



5.4.2.5 Verify Output

1. To start the scanning and communication process, click Start.

Figure 5-25. Start Communication

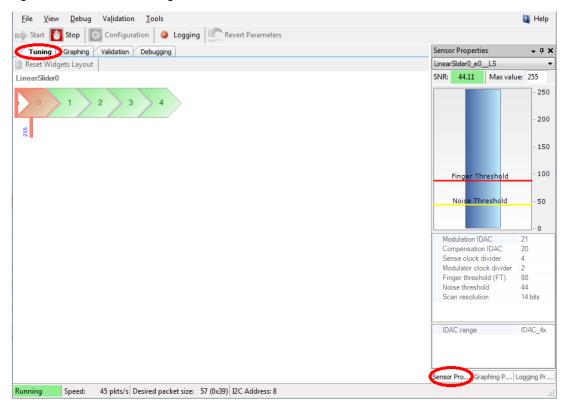




 Select a sensor in the Tuning tab. A red outline is seen on the selected sensor. Different CapSense parameters are shown in Sensor Properties tab on the bottom-right. You cannot edit the settings because auto-tuning is used in this project; auto-tuning automatically sets all the parameters. Touch the selected sensor and observe the response in the tuner window.

Note: The board is designed according to layout guidelines for CapSense (best practices) for 1.5-mm overlay. Therefore, it is recommended that an overlay (not shipped with the kit) be used while using the CapSense code example with tuning.

Figure 5-26. Sensor Tuning



 In the Graphing tab, the CapSense results: Raw counts, Baseline, Signal (difference count) and On/Off status for each sensor are represented as a graph. Click the Graphing Properties tab on the bottom-right to select the slider element for which the CapSense results are to be shown in graphing tab.



4. Select the sensor parameters to observe, as shown in the following figure. The graph of the selected parameters is shown.

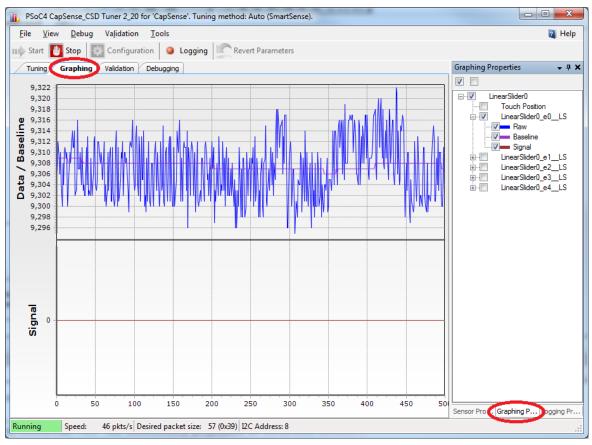
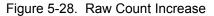
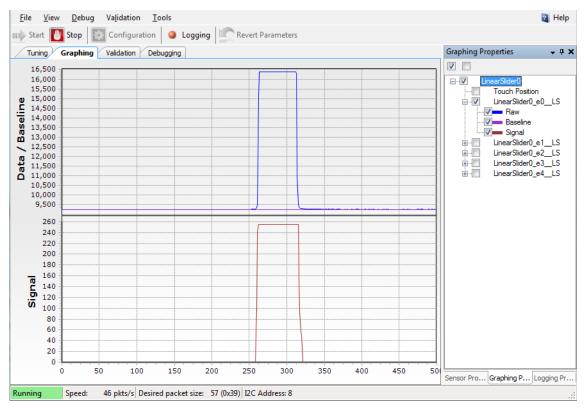


Figure 5-27. Sensor Parameter Graph



5. Touch a sensor or slider element and see the increase in raw counts.







6.1 Using PSoC 5LP as USB-UART Bridge

6. Advanced Topics

The PSoC 5LP serves as a USB-UART bridge, which can communicate with the COM terminal software. This section explains how to create a PSoC 4 code example to communicate with the COM terminal software. This project is available with other code examples for the PSoC 4 Pioneer Kit at the element14 web page, 100 Projects in 100 days.

Users who have a Windows operating system that does not have HyperTerminal can use an alternate terminal software such as PuTTY.

1. Create a new project targeting the PSoC 4 device in PSoC Creator. Select an appropriate location for your project and rename the project as required.

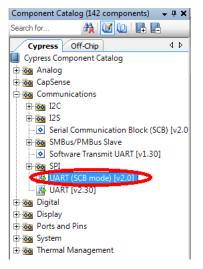
Figure 6-1. Opening New Project from PSoC Creator

New Project		8	X
Design Other			4 ۵
 Default Templates 			*
PSoC 3 Design	1	Creates a PSoC 3, 8-bit 8051, design project.	
PSoC 4000 De	sian	Creates a PSoC 4000, 32-bit ARM Cortex-M0, design project.	
PSoC 4100 / P	SoC 4200 Design	Creates a PSoC 4100 / PSoC 4200, 32-bit ARM Cortex-M0, design project.	
PSoC 4100 BL	E / PSoC 4200 BLE Design	Creates a PSoC 4100 BLE / PSoC 4200 BLE, 32-bit ARM Cortex-M0, design project.	=
PRoC BLE Desi	ign	Creates a PRoC BLE, 32-bit ARM Cortex-M0, design project.	
PSoC 4100M /	/ PSoC 4200M Design	Creates a PSoC 4100M / PSoC 4200M, 32-bit ARM Cortex-M0, design project.	
PSoC 5LP Desi	ign	Creates a PSoC 5LP, 32-bit ARM Cortex-M3, design project.	_
PSoC 3 Starter Design:	S		
PSoC 4000 Starter Des	signs		
■ PSoC 4100 / PSoC 42	00 Starter Designs		
PSoC 4100 BLE / PSoC	C 4200 BLE Starter Designs		-
Name:	Design01		
Location:	C:\Users\srds\Documents		
Device:	CY8C4245AXI-483 - (Default	PSoC 4100 / PSoC 4200 Device)	
Workspace:	Create New Workspace	Ψ.	
Workspace name:	Design01		
Project template:	Pre-populated schematic	•	
		OK Cance	ł



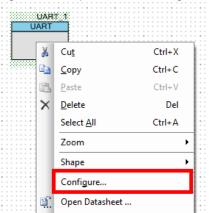
2. Drag and drop a UART (SCB) component to the top design.

Figure 6-2. UART Component Under Component Catalog



3. To configure the UART, double-click or right-click on the UART component and select **Configure**.







4. Configure the UART as shown in the following figures.

Figure 6-4. UART Configuration Window

Configure 'SCB_P4'	? ×
Name: UART_1	
Configuration UART Basic UART Advanced Built-in	٩ ۵
O Unconfigured SCB	
© 12C	
© EZI2C	
© SPI	
O UART	
Datasheet OK Apply	Cancel

Figure 6-5. UART Basic Configuration Window

Configure 'SCB_P4'		? <mark>X</mark>
Name: UART		
Configuration	RT Basic UART Advanced Built-in	4 Þ
Mode: Star	andard 👻	
Direction: TX	(+ RX ▼	
Baud rate (bps): 960	Actual baud rate (bps): UNKNOWN (Press "Apply")	
Data bits: 8 bi	its 🔹	
Parity: Nor	ne 💌	
Stop bits: 1 bi	it 💌	
Oversampling: 12	<u>A</u>	
Clock from terminal		
Median filter		
Retry on NACK		
Inverting RX		
Enable wakeup from D	Deep Sleep Mode	
Low power receiving		
Datasheet	OK Apply C	ancel
		.:1



Configuration UART Basic UART Basic UART Basic UART Basic UART Buffers size	ART Advanced Built-in Interrupt	4 Þ
RX buffer size: 8	None	
TX buffer size: 8 🚖	O Internal	
Byte mode	External	
Interrupt sources		
UART done	RX FIFO not empty	
TX FIFO not full	RX FIFO full	
TX FIFO empty	RX FIFO overflow	
TX FIFO overflow	RX FIFO underflow	
TX FIFO underflow	RX frame error	
TX lost arbitration	RX parity error	
TX NACK	RX FIFO level: 7 -	
TX FIFO level: 0		
Multiprocessor mode	RX FIFO drop	
Address (hex): 2	On parity error	
Mask (hex): FF 🔺	On frame error	
Accept matching address in RX FIF	C	
Flow control		
RTS Polarity: Active Low	▼ RTS FIFO level: 4	
CTS Polarity: Active Low		
CTS Polanty: Active Low	*	

Figure 6-6. UART Advanced Configuration Window

- 5. Click **Apply** and then **OK** to save the changes made to UART configuration.
- 6. Select P0[4] for UART RX and P0[5] for UART TX in the Pins tab of <Project.cydwr>.

Figure 6-7. Pin Selection

 ↓ ↓ ↓ ↓				⊳×		
Alias	Alias Name / Port		Pin		Lock	
	\UART:rx\	P0[4] SCB1:uart_rx, SCB1:i2c_scl, SCB1:spi mosi	•	28	•	V
	\UART:tx\	P0[5] SCB1:uart_tx, SCB1:i2c_sda, SCB1:api_miso	-	29	•	V



7. Place the following code in your *main.c* project file. The code will echo any UART data received.

```
int main()
{
    uint8 ch;
    /* Start SCB UART TX+RX operation */
    UART_Start();
/* Transmit String through UART TX Line */
UART_UartPutString("CY8CKIT-042 USB-UART");
    for(;;)
    {
        /* Get received character or zero if nothing has been received yet
* /
        ch = UART_UartGetChar();
        if(0u != ch)
        {
 /* Send the data through UART. This functions is blocking and waits until
there is an entry into the TX FIFO. */
            UART_UartPutChar(ch);
        }
    }
}
```



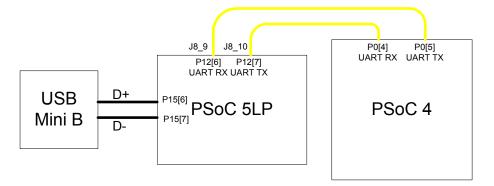
 Build the project by clicking Build > Build {Project Name} or [Shift] + [F6]. After the project is built without errors and warnings, program (by clicking Debug > Program) the project to PSoC 4 through the PSoC 5LP USB programmer or MiniProg3.

Connect the RX line of the PSoC 4 to J8_10 and TX line of the PSoC 4 to J8_9, as shown in the following figures.

Figure 6-8. UART Connection Between PSoC 4 and PSoC 5LP



Figure 6-9. Block Diagram of UART Connection Between PSoC 4 and PSoC 5LP



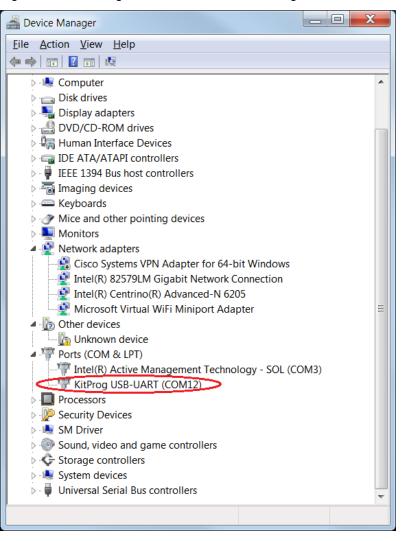
Note: UART RX and UART TX can be routed to any digital pin on PSoC 4 based on the configuration of the UART component. An SCB implementation of UART will route the RX and TX pins to either one of the following subsets: (P0[4], P0[5]) or (P3[0],P3[1]) or (P4[0],P4[1]).



To communicate with the PSoC 4 from the terminal software, follow this procedure:

 Connect USB Mini B to J10. The kit enumerates as a KitProg USB-UART and is available under the Device Manager, Ports (COM & LPT). A communication port is assigned to the KitProg USB-UART.

Figure 6-10. KitProg USB-UART in Device Manager





2. Open HyperTerminal and select **File > New Connection** and enter a name for the new connection and click **OK**.

For PuTTY, double-click the putty icon and select **Serial** under **Connection**.

Figure 6-11. Open New Connection

HyperTerminal

Connection Description	? ×
New Connection	
Enter a name and choose an icon for the connection:	
Name:	
USB-UART communication	
lcon:	
< III	4
ОК	Cancel

PuTTY

Real PuTTY Configuration			
Category:			
Category: Session Logging Terminal Keyboard Bell Features Window Appearance Behaviour Translation Selection Colours Connection Data Proxy Telnet	Basic options for your PuTTY session Specify the destination you want to connect to Host Name (or IP address) Port 22 Connection type: Raw I elnet Rlogin SSH Save or delete a stored session Save d Sessions Default Settings Load		
Rlogin B SSH Serial	Delete Close window on exit Always Never Open Cancel		



3. A new window opens, where the communication port can be selected.

In HyperTerminal, select **COMX** (or the specific communication port that is assigned to KitProg USB-UART) in **Connect using** and click **OK**.

In PuTTY enter the COMX in Serial line to connect to.

This code example uses COM12.

Figure 6-12. Select Communication Port

HyperTerminal

Connect To	? ×
USB-UA	RT communication
Enter details for t	he phone number that you want to dial:
Country/region:	India (91) 💌
Ar <u>e</u> a code:	080
Phone number:	
Co <u>n</u> nect using:	COM12 -
	OK Cancel

PuTTY

RuTTY Configuration		X
Category:		
Category: 	Options controlling Select a serial line Serial line to connect to Configure the serial line Speed (baud) Data bits Stop bits Parity Elow control	g local serial lines COM12 9600 8 1 1 None • XON/XOFF •
About		Open <u>C</u> ancel



4. In HyperTerminal, select 'Bits per second', 'Data bits', 'Parity', 'Stop bits', and 'Flow control' under **Port Settings** and click **OK**.

Make sure that the settings are identical to the UART settings configured for PSoC 4.

In PuTTY select 'Speed (baud)', 'Data bits', 'Stop bits', 'Parity' and 'Flow control' under **Configure the serial line**. Click **Session** and select **Serial** under **Connection type**.

Serial line shows the communication port (COM12) and **Speed** shows the baud rate selected. Click **Open** to start the communication.

Figure 6-13. Configure the Communication Port

HyperTerminal

COM12 Properties	₽ ×
Port Settings	
<u>B</u> its per second:	9600 💌
<u>D</u> ata bits:	8 🔻
<u>P</u> arity:	None
Stop bits:	1 •
<u>F</u> low control:	None
	Restore Defaults
0	K Cancel <u>A</u> pply

PuTTY

Real PuTTY Configuration		X	
Category:			
Category: Session Logging Terminal Feelbaures Window Appearance Behaviour Translation Selection Connection Data Proxy Telnet Rlogin SSH Serial	Options controlling loca Select a serial line Serial line to connect to Configure the serial line Speed (baud) Data bits Stop bits Parity Elow control	I serial lines COM12 9600 8 1 None • None •	
About	Open	<u>C</u> ancel	



RuTTY Configuration	X
Category:	
Category. - Cogging - Terminal - Keyboard - Bell - Features - Window - Appearance - Behaviour - Translation - Selection - Colours - Connection - Data - Proxy - Telnet - Rlogin - SSH - Serial	Basic options for your PuTTY session Specify the destination you want to connect to Serial line Speed COM12 9600 Connection type: Image: Connection type: Raw Image: Connection type: Raw Image: Connection type: Raw Image: Connection type: Raw Image: Connection type: Connection type: Image: Connection type: Default Settings Load Default Settings Load Save Image: Connection type: Always Never Only on clean exit
About	Open <u>C</u> ancel

Figure 6-14. Select Communication Type in PuTTY

 Enable Echo typed characters locally under File > Properties > Settings > ASCII Setup, to display the typed characters on HyperTerminal. In PuTTY, enable the Force on under Terminal > Line discipline options to display the typed characters on the PuTTY.

Figure 6-15. Enabling Echo of Typed Characters in HyperTerminal

ASCII Setup			
ASCII Sending			
Send line ends with line feeds			
Echo typed characters locally			
Line delay: 0 milliseconds.			
Character delay: 0 milliseconds.			
ASCII Receiving Append line feeds to incoming line ends Eorce incoming data to 7-bit ASCII Wrap lines that exceed terminal width OK			

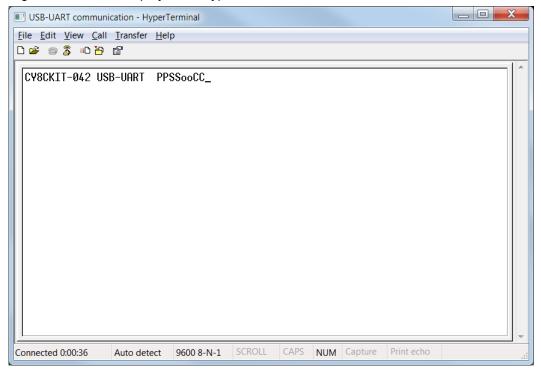


🔀 PuTTY Configuration	X
PuTTY Configuration ategory: - Session - Logging - Terminal - Keyboard - Bell - Features - Window - Appearance - Behaviour - Translation - Selection	Options controlling the terminal emulation Set various terminal options Auto wrap mode initially on DEC Origin Mode initially on Implicit CR in every LF Implicit LE in every CR Use background colour to erase screen Enable blinking text Answerback to ^E:
	Answerback to "E: PuTTY Line discipline options Local echo: Auto Local line ediţing: Auto Force on Force off Force off
	Remote-controlled printing Printer to send ANSI printer output to: None (printing disabled)
About	<u>O</u> pen <u>C</u> ancel

Figure 6-16. Enabling Echo of Typed Characters in PuTTY

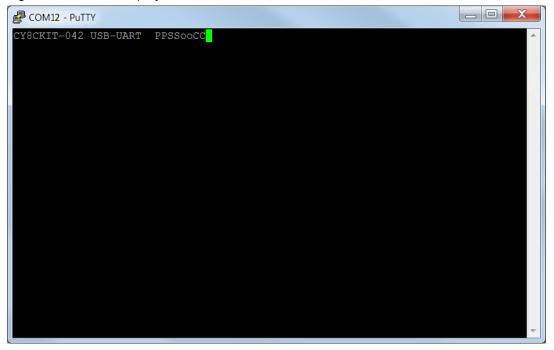
6. The COM terminal software displays both the typed data and the looped back data from the PSoC 4 UART.

Figure 6-17. Data Displayed on HyperTerminal











6.2 Using PSoC 5LP as USB-I2C Bridge

The PSoC 5LP serves as a USB-I2C bridge, which can be used to communicate with the USB-I2C software running on the PC. This project is available with other code examples for the PSoC 4 Pioneer Kit at the element14 web page, 100 Projects in 100 days.

The following steps describe how to use the USB-I2C bridge, which can communicate between the BCP and the PSoC 4.

1. Create a new project targeting the PSoC 4 device in PSoC Creator.

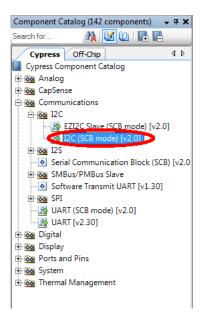
Figure 6-19. Opening a New Project in PSoC Creator

lew Project		8	×
Design Other			٩ ۵
 Default Templates 			_
PSoC 3 Desig	n	Creates a PSoC 3, 8-bit 8051, design project.	
PSoC 4000 D	esian	Creates a PSoC 4000, 32-bit ARM Cortex-M0, design project.	
PSoC 4100 /	PSoC 4200 Design	Creates a PSoC 4100 / PSoC 4200, 32-bit ARM Cortex-M0, design project.	
PSoC 4100 BI	LE / PSoC 4200 BLE Design	Creates a PSoC 4100 BLE / PSoC 4200 BLE, 32-bit ARM Cortex-M0, design project.	E
PRoC BLE Des	sign	Creates a PRoC BLE, 32-bit ARM Cortex-M0, design project.	
	/ PSoC 4200M Design	Creates a PSoC 4100M / PSoC 4200M, 32-bit ARM Cortex-M0, design project.	
PSoC 5LP Des	2	Creates a PSoC 5LP, 32-bit ARM Cortex-M3, design project.	
PSoC 3 Starter Design			
PSoC 4000 Starter De	-		
	2		
PSoC 4100 BLE / PSo	C 4200 BLE Starter Designs		-
Name:	Design01		
Location:	C:\Users\srds\Documents		
Device: CY8C4245AXI-483 - (Default		PSoC 4100 / PSoC 4200 Device)	
Workspace:	Create New Workspace	Ψ	
Workspace name:	Design01		
Project template:	Pre-populated schematic	•	
		OK Cancel	

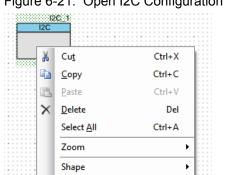


2. Drag and drop an I2C component to the top design.

Figure 6-20. I2C Component in Component Catalog



3. To configure the I2C component, double-click or right-click on the I2C component and select **Configure**.



Configure...

Figure 6-21. Open I2C Configuration Window



4. Configure the I2C with the following settings.

Figure 6-22. I2C Configuration Tab

Configure 'SCB_P4'	? ×
Name: I2C	
	4 ۵
Configuration I2C Basic I2C Advanced Built-in Unconfigured SCB	<u> </u>
 Incomigated Set I2C 	
© EZI2C	
© SPI	
O UART	
- Over	
	Grand
Datasheet OK Apply	Cancel



Figure 6-23. I2C Basic Tab

onfigure 'SCB_P4'	? ×
Name: I2C	
Configuration I2C Basic I2C Advanced Built-in	٩ ۵
Mode: Slave	
Data rate (kbps): 400 🔹 Actual data rate (kbps): UNKNOWN (Press "Apply")	
Oversampling factor: 16 🗼 Low: 8 🗼 High: 8 🛓	
Manual oversample control	
Clock from terminal	
Byte mode	
Address R/W	
Slave address (7-bits): 0x08 0 0 0 1 0 0 X	
Slave address mask: 0xFE MSB 1 1 1 1 1 1 1 0	
Accept matching address in RX FIFO	
Enable wakeup from Deep Sleep Mode	
Datasheet OK App	ly Cancel

Figure 6-24. I2C Advanced Tab

Configure 'SCB_P4'	? ×
Name: I2C	
Configuration I2C Basic I2C Advanced Built-in	4 ⊳
Slew rate: Fast I2C bus voltage (V): 3.3	
Datasheet OK A	pply Cancel



- 5. Click **Apply** and then **OK** to save the changes.
- 6. Select pin P3[0] for the I2C SCL and pin P3[1] for the I2C SDA in the Pins tab of <poject.cydwr>.

Figure 6-25. Pin Selection

Alias	Name 🗠	Port	Pi	in	Lock		
	\I2C:scl\	P3[0] TCPWM0:line_out, SCB1:uart_rx, SCB1:i2c scl, SCB1:spi mosi	11	•			
	\I2C:sda\	P3[1] TCPWM0:line_out_compl, SCB1:uart_tx, SCB1:i2c sda, SCB1:spi miso	12	•			

7. Place the following code in your *main.c* project file. The code will enable the PSoC 4 device to transmit and receive I2C data to and from the BCP application.

```
int main()
ł
uint8 wrBuf[10]; /* I2C write buffer */
uint8 rdBuf[10]; /* I2C read buffer */
uint8 indexCntr;
uint32 byteCnt;
/* Enable the Global Interrupt */
CyGlobalIntEnable;
/* Start I2C Slave operation */
I2C Start();
/* Initialize write buffer */
I2C_I2CSlaveInitWriteBuf((uint8 *) wrBuf, 10);
/* Initialize read buffer */
I2C_I2CSlaveInitReadBuf((uint8 *) rdBuf, 10);
for(;;) /* Loop forever */
{
/* Wait for I2C master to complete a write */
if(Ou != (I2C_I2CSlaveStatus() & I2C_I2C_SSTAT_WR_CMPLT))
{
      /* Read the number of bytes transferred */
      byteCnt = I2C_I2CSlaveGetWriteBufSize();
      /* Clear the write status bits*/
      I2C_I2CSlaveClearWriteStatus();
/* Move the data written by the master to the read buffer so that the
      master can read back the data */
      for(indexCntr = 0; indexCntr < byteCnt; indexCntr++)</pre>
```



```
{
rdBuf [indexCntr] = wrBuf[indexCntr]; /* Loop back the data to the read
            buffer */
      }
/* Clear the write buffer pointer so that the next write operation will
      start from index 0 */
      I2C I2CSlaveClearWriteBuf();
/* Clear the read buffer pointer so that the next read operations starts
      from index 0 */
      I2C I2CSlaveClearReadBuf();
}
/* If the master has read the data , reset the read buffer pointer to 0
and clear the read status */
if(Ou != (I2C_I2CSlaveStatus() & I2C_I2C_SSTAT_RD_CMPLT))
{
/* Clear the read buffer pointer so that the next read operations starts
from index 0 */
      I2C_I2CSlaveClearReadBuf();
      /* Clear the read status bits */
      I2C_I2CSlaveClearReadStatus();
}
}
}
```

- Build the project by clicking Build > Build Project or [Shift]+[F6]. After the project is built without errors and warnings, program ([Ctrl]+[F5]) this code onto the PSoC 4 through the PSoC 5LP programmer or MiniProg3.
- 7. Open the BCP from Start > All Programs > Cypress > Bridge Control Panel <version number>.
- 8. Connect to KitProg/ under Connected I2C/SPI/RX8 Ports.



Windge Control Panel		
<u>File Editor Chart Execu</u>	te <u>I</u> ools <u>H</u> elp	
🖻 🛛 🗑 🏚 🖻 🖉 🧮 🛛		
Editor Chart Table File		
Opening Port Successfully Conn KitProg Version 2		
	Connected I2C/SPI/RX8 Ports	Power Protocol
🔇 Reset 🖹 🛣 List	Send Send all strings: KiProg/1A19172E03242400 Repeat count COM3 COM3	○ +5.0V ● I2C
🔳 Stop 🐺 Repeat 🛔	To file Scan paried me:	
	Scan period, ms: 0 v	© +1.8V
1:1 Syntax:OK	Connected Powered Voltage: 3327 mV	

Figure 6-26. Connecting to KitProg/ in BCP

9. Open **Protocol Configuration** from the **Tools** menu and select the appropriate **I2C Speed**. Make sure the I2C speed is the same as the one configured in the I2C component. Click **OK** to close the window.

Figure 6-27. Opening Protocol Configuration Window in BCP

ОК

Cancel

rigure o 27. opennig i lotoo	or conligaration wi		501		
🗱 Bridge Control Panel					
<u>File E</u> ditor <u>C</u> hart E <u>x</u> ecute <u>T</u>	ools <u>Help</u>				
🍺 🖩 🚊 🕼 🛍 🖉 🧮 🗮 🧲	Protocol Configuration	F7			
Editor Chart Table File	I2C <u>B</u> ootloader	F3			
SPI I2C RX8 (UAR T) I2C Speed 1 MHz 400 kHz 100 kHz	Hz 💿 50 kHz				



10. From the BCP, transfer five bytes of data to the I2C device with slave address 0x08. The log shows whether the transaction was successful. A '+' indication after each byte indicates that the transaction was successful and a '-' indicates that the transaction was a failure.

Figure 6-28.	Entering	Commands	in BCP
--------------	----------	----------	--------

🖡 Bridge Control Panel 💦 🔤	- 0 X
File Editor Chart Execute Tools Help	
Editor Chant Table Fie	
Generates Stop condition on I2C bus Data bytes Slave address	•
"write data" command	
Indicates Acknowledgement (ACK) Select Port in the PortList, then try to connect	F
Opening Port Successfully Connected to KitProg/1A19172E03242400 KitProg Version 2.02 w 08⊕ 00+ 01+ 02+ 03+ 04+ p	Ŧ
	Þ
1:21 Syntax:OK ok Connected Powered Voltage: 3325 mV	



5		
🗱 Bridge Control Panel		_ 0 ×
Eile Editor Chart Execute Tools Help		
☞ ■ ⑧ 圖 № ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■		
Editor Chart Table File		
w 7 00 01 02 03 04 p		*
Indicates Not acknowledged (NACK)		• •
Select Port in the PortList, then try to connect Opening Port Successfully Connected to KitProg/1A19172E03242400 KitProg Version 2.02 w 08+ 00+ 01+ 02+ 03+ 04+ p w 07- 00- 01- 02- 03- 04- p		
		÷
Ø Reset	Power Protocol 0 +50V 0 12C -33V 0 SPI 0 -25V 0 RXS (UART)	
1:21 Syntax: OK ok Connected Powered Voltage: 3327 mV		

Figure 6-29. NACK Indication in BCP

11. From the BCP, read five bytes of data from the I2C slave device with slave address 0x08. The log shows whether the transaction was successful.

🗱 Bridge Control Panel	_ 0 X
Eile Editor Chart Execute Iools Help	
● B A A A A A A A A A A A A A A A A A A	
Editor Chast Table File	
Generates Stop condition on 12C bus No. of data bytes to read x - Reserved symbol, which means that 1 byte of data should be read Slave address	
"read data" command Data bytes read from the slave device	*
	P.
Select Port in the PortList, then try to connect Opening Port Successfully Connected to KitProg/lA19172E03242400 KitProg Version 2.02 w 08+ 00+ 01+ 02+ 03+ 04+ p w 07- 00- 01- 02- 03- 04- p r 08+ 00+ 01+ 02+ 03+ 04+ p	
	- F
Owner service Owner service Protocol Protocol	

Figure 6-30. Read Data Bytes from the BCP

Note: Refer Help Contents under Help in BCP or press [F1] for details of I2C commands.



6.3 Developing Applications for PSoC 5LP

The PSoC 4 Pioneer Kit has an onboard PSoC 5LP whose primary function is that of a programmer and a bridge. You can build either a normal project or a bootloadable project using the PSoC 5LP.

The PSoC 5LP connections in the Pioneer board are summarized in Figure 6-31. J8 is the I/O connector (see section 4.3.7 PSoC 5LP GPIO Header (J8)). The USB (J10) is connected and used as the PC interface. But you can still use this USB connection to create customized USB designs.

The programming header (J7) is meant for standalone programming. This header needs to be populated. See the 'No Load Components' section in A.6 Bill of Materials (BOM) on page 121.

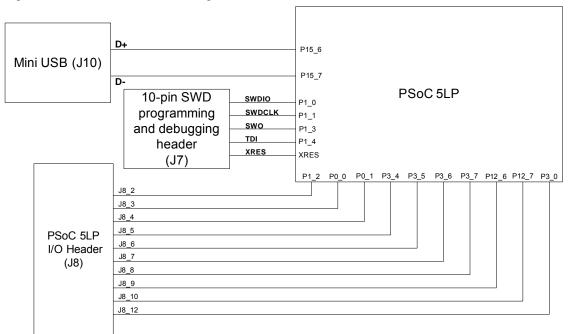


Figure 6-31. PSoC 5LP Block Diagram

6.3.1 Building a Bootloadable Project for PSoC 5LP

All bootloadable applications developed for the PSoC 5LP should be based on the bootloader hex file, which is programmed onto the kit. The bootloader hex file is available in the kit files or can be downloaded from the kit web page.

The hex files are included in the following kit installer directory: <Install_Directory>\CY8CKIT-042 PSoC 4 Pioneer Kit\ <version>\Firmware\Programmer\KitProg_Bootloader

Figure 6-32. KitProg Bootloader Hex File Location

	Firmware • Programmer • KitProg_Bootloader	✓ 4 Search
Organize 🝷 Includ	e in library Share with Burn New	folder
★ Favorites	Name	Date modified Type
💻 Desktop	KitProg_Bootloader.elf	3/18/2013 6:38 PM ELF File
bownloads	KitProg_Bootloader.hex	3/18/2013 6:38 PM HEX File
💝 Dropbox 🦷		
Recent Places		



To build a bootloadable application for the PSoC 5LP, follow this procedure:

 In PSoC Creator, choose New > Project and click PSoC 5LP Design; select Launch Device Selector from the drop-down list for Device to bring up the Select PSoC 5LP Device window and select CY8C5868LTI-LP039, as shown in Figure 6-33. Click OK.

Figure 6-33.	Opening	New Project	in PSoC Creator

New Project		2 ×
Design Other		٩ ۵
Default Templates		▲
PSoC 3 Design	1	Creates a PSoC 3, 8-bit 8051, design project.
PSoC 4000 De	sign	Creates a PSoC 4000, 32-bit ARM Cortex-M0, design project.
PSoC 4100 / P	SoC 4200 Design	Creates a PSoC 4100 / PSoC 4200, 32-bit ARM Cortex-M0, design project.
PSoC 4100 BL	E / PSoC 4200 BLE Design	Creates a PSoC 4100 BLE / PSoC 4200 BLE, 32-bit ARM Cortex-M0, design project.
PRoC BLE Desi	2	Creates a PRoC BLE, 32-bit ARM Cortex-M0, design project.
	(PSoC 4200M Design	Creates a PSoC 4100M / PSoC 4200M, 32-bit ARM Cortex-M0, design project.
PSoC 5LP Desi		Creates a PSoC 5LP, 32-bit ARM Cortex-M3, design project.
PSoC 3 Starter Designs		
PSoC 4000 Starter Des	2	
PSoC 4100 BLE / PSoC	C 4200 BLE Starter Designs	*
Name:	Design01	
Location:	C:\Users\srds\Documents	
Device:	CY8C5868AXI-LP035	•
Workspace:	Create New Workspace	Ţ
Workspace name:	Design01	
Project template:	Empty schematic	·
		OK Cancel
		h.

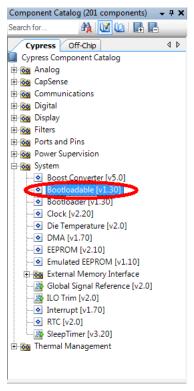


Igure 6-34. S		•					101									-		x
Select PSoC 5LP Device (Other architectures available after project creation)																		
🕑 View Datasheet 羘	Hide/Show (Columns	Reset to I	Defau	ts													
(CPU		Package	Max Frequency (MHz)	Flash (KB)	SRAM (KB)	EEPROM (bytes)	IO	CapSense	Bluetooth	LCD Drive (mux ratio)	Timer/Counter/PWM	Communication Blocks	UDB	Pre-configured Blocks	USB	CAN 2.0b	
Filters:						_		_						_		_	_	
CY8C5868LTI-LP036	ARM CM3	PSoC 5LP	68-QFN	67	256	64	2	46	Y		~	4	1	24	-	-	-	
CY8C5868LTI-LP038	ARM CM3	PSoC 5LP	68-QFN	67	256	64	2	48	Y		√	4	1	24	-	FS	-	
CY8C5868LTI-LP039	ARM CM3	PSoC 5LP	68-QFN	67	256	64	2	48	Y		~	4	1	24	-	FS	1]
CY8C5888AXI-LP096	ARM CM3	PSoC 5LP	100-TQFP	80	256	64	2	72	Y		~	4	1	24	-	FS	1	
CY8C5888AXQ-LP096	ARM CM3	PSoC 5LP	100-TQFP	80	256	64	2	72	Y		✓	4	1	24	-	FS	1	+
•		III															۴	
64 of 64 devices found	Clear Filters	i																
													ОК			Can	cel	

Figure 6-34. Selecting Device in PSoC Creator

2. Navigate to the Schematic view and drag and drop a bootloadable component on the top design.

Figure 6-35. Bootloadable Component in Component Catalog





Set the dependency of the Bootloadable component by selecting the **Dependencies** tab in the configuration window and clicking the **Browse** button. Select the *KitProg_Bootloader.hex* and *KitProg_Bootloader.elf* files; click **Open**.

Figure 6-36.	Configuration	Window d	of Bootloadable C	omponent

Configure 'Bootloadable'	? X
Name: Bootloadable_1	
General Dependencies Built-in	4 ۵
Bootloadable projects require a reference to the associated Bootloader project's HEX files. The HEX files extension is *.hex. The ELF files extension depends on IDE and c *.elf, *.out, *.axf, or other.	
Bootloader HEX file:	
Bootloader ELF file:	wse
Bro	wse
Datasheet OK Apply	Cancel

Figure 6-37. Selecting KitProg Bootloader Hex File

Select a Bootloader Hex File			×
Coover II. « Firmwar	• Programmer • KitProg_Bootloader	▼ 🍫 Searc	ch KitProg_Bootloa 🔎
Organize New fold	er		≣ - □ 0
💻 Desktop 🖍	Name	Date modified	Туре
Downloads Oropbox	KitProg_Bootloader.hex	3/18/2013 6:38	HEX File
S Recent Places			
 ☐ Libraries ☐ Documents ④ Music ④ Pictures ④ Videos 			
ille Computer			
Windows7_OS (C: Control Control Contr			
	KitProg_Bootloader.hex	 Hex Files (Open 	(*.hex)



<u> </u>		
Select a Bootloader Hex File		×
G V Firmware > Pr	ogrammer 🕨 KitProg_Bootloader	✓ Search KitProg_Bootloader
Organize 🔻 New folder		:= - 1 🔞
 ★ Favorites ■ Desktop Downloads Recent Places ■ Libraries ■ Documents → Music ■ Pictures ▼ Videos 	Name KitProg_Bootloader.elf	Date modified Type 4/18/2013 1:07 AM ELF File
Computer Local Disk (C:) CY8C58 Family Processo File <u>n</u> ame:	- <	

Figure 6-38. Selecting KitProg Bootloader Elf File

3. In the **General** tab, check the **Manual application image placement** checkbox and set the **Placement address** as "0x00002800" as shown in Figure 6-39.

onfigure 'Bootloadable'		? <mark>×</mark>
Name: Bootloadable	,1	
General Depen	dencies Built-in	4 ۵
Application version:	0x0000	
Application ID:	0x0000	
Application custom ID:	0x0000000	
 Manual application in Placement address: 		
Placement address:	0x00002800	
Datasheet	OK A	ply Cancel

Figure 6-39. Bootloadable Component-General Tab

4. Develop your custom project.



5. The NVL setting of the Bootloadable project and the KitProg_Bootloader project must be the same. The *KitProg_Bootloader.cydwr* system settings is shown in the following figure.

Figure 6-40. KitProg Bootloader System Settings

ption	Value
Configuration	
- Device Configuration Mode	Compressed
- Enable Error Correcting Code (ECC)	10 m
Store Configuration Data in ECC Memory	
- Instruction Cache Enabled	12 I
Enable Fast IMO During Startup	2
- Unused Banded ID	Allow with info
- Heap Size (bytes)	0×1000
Stack Size (bytes)	0x4000
Include CMSIS Core Peripheral Library Files	×
Programming\Debugging	
	GPIO
- Enable Device Protection	E
Embedded Trace (ETM)	5
Use Optional XRES	5
Operating Conditions	
- VDDA (V)	5.0
- Variable VDDA	0
- VDDD (V)	5.0
- VDBI08 (V)	5.0
- VDB01 (V)	5.0
- VD02 (V)	5.0
	5.0
- VDI03 (V)	

- 6. Build the project in PSoC Creator by selecting Build > Build Project or [Shift]+[F6].
- To download the project on to the PSoC 5LP device, open the Bootloader Host Tool, which is available from PSoC Creator. Select Tools > Bootloader Host.

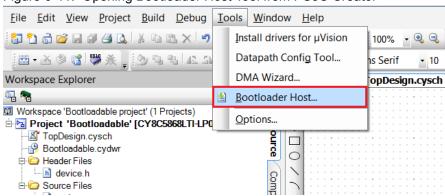


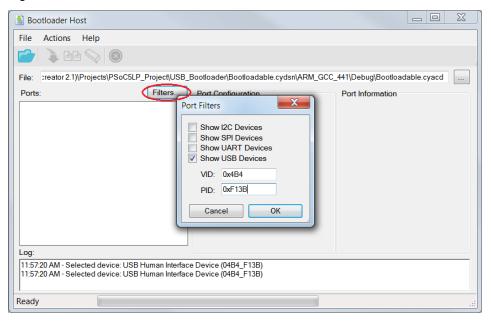
Figure 6-41. Opening Bootloader Host Tool from PSoC Creator

Keep the reset switch (SW1) pressed and plug in the USB Mini-B connector. If the switch is
pressed for more than 100 ms, the PSoC 5LP enters into bootloader. The PSoC 5LP also enters
into bootloader when the power supply jumper for the PSoC 4 (J13) is removed and
subsequently the USB Mini-B connector is plugged into header J10.



9. In the Bootloader Host tool, click **Filters** and add a filter to identify the USB device. Set VID as **0x04B4**, PID as **0xF13B**, and click **OK**.

Figure 6-42. Port Filters Tab in Bootloader Host Tool



10. In the Bootloader Host tool, click the **Open File** button to browse to the location of the bootloadable file (*.cyacd).

🛓 Bootloader Host		
<u>File</u> <u>A</u> ctions <u>H</u> elp		
🖆 🔰 BB 📎 🛞		
Fle: C:\Jsers\ancy\Desktop\Bootloadable project	\Bootloadable.cydsn\CortexM3\ARM_GCC	441\Debug\Bootloadable.cyacd
Forts: Filters	Port Configuration USB Voconfiguration necessary for this port.	Port Information VID: 0484 PID: F138
Log: 12:35:02 PM - Selected device: USB Human Interface 12:35:02 PM - Selected device: USB Human Interface 12:35:08 PM - Selected device: USB Human Interface	e Device (04B4_F13B)	

Figure 6-43. Opening Bootloadable File from Bootloader Host Tool



11. Press the **Program** button in the Bootloader Host tool to program the device.

Figure 6-44. Selecting Bootloadable .cyacd File from Bootloader Host

🛓 Open			X
CortexM3	► ARM_GCC_441 ► Debug ►	🕶 🍫 Search Debug	Q
Organize New folde		• •	
★ Favorites	Name	Date modified	Туре
💻 Desktop	📙 .deps	4/18/2013 12:34 PM	File folder
🐌 Downloads	Bootloadable.cyacd	4/18/2013 12:34 PM	CYACD File
 ➡ Dropbox ➡ Recent Places ➡ Libraries ➡ Documents ➡ Music ➡ Pictures ➡ Videos 			
ik Computer	•		
🜬 Windows7 OS (C: 🍸			
File <u>n</u> arr	e: Bootloadable.cyacd -	bootioader mes	▼ Cancel

12. If bootload is successful, the log of the tool displays "Successful"; otherwise, it displays "Failed" and a statement for the failure.

Notes:

- The PSoC 5LP pins are brought to the PSoC 5LP GPIO header (J8). These pins are selected to support high-performance analog and digital projects. See A.2 Pin Assignment Table on page 116 for pin information.
- Take care when allocating the PSoC 5LP pins for custom applications. For example, P2[0]–P2[4] are dedicated for programming the PSoC 4. Refer to A.1 CY8CKIT-042 Schematics on page 113 before allocating the pins.
- 3. When a normal project is programmed onto the PSoC 5LP, the initial capability of the PSoC 5LP to act as a programmer, USB-UART bridge, or USB-I2C bridge in not available.
- 4. The status LED does not function unless used by the custom project.

For additional information on bootloaders, refer to Cypress application note, AN73503 - USB HID Bootloader for PSoC 3 and PSoC 5LP.



6.3.2 Building a Normal Project for PSoC 5LP

A normal project is a completely new project created for the PSoC 5LP device on the CY8CKIT-042. Here the entire flash of the PSoC 5LP is programmed, overwriting all bootloader and programming code. To recover the programmer, reprogram the PSoC 5LP device with the factory-set *KitProg.hex* file, which is shipped with the kit installer.

The *KitProg.hex* file is available at the following location:

```
<Install_Directory>\CY8CKIT-042 PSoC 4 Pioneer Kit\<version>\Firm-
ware\Programmer\KitProg
```

This advanced functionality requires a MiniProg3 programmer, which is not included with this kit. The MiniProg3 can be purchased from www.cypress.com/go/CY8CKit-002.

To build a normal project for the PSoC 5LP, follow these steps:

 In PSoC Creator, choose New > Project and click PSoC 5LP Design; select Device as CY8C5868LTI-LP039 (see Figure 6-45), and then click OK.

New Project		8	X
Design Other			4 ۵
 Default Templates 			•
PSoC 3 Design	n	Creates a PSoC 3, 8-bit 8051, design project.	
PSoC 4000 D	esign	Creates a PSoC 4000, 32-bit ARM Cortex-M0, design project.	
PSoC 4100 / I	PSoC 4200 Design	Creates a PSoC 4100 / PSoC 4200, 32-bit ARM Cortex-M0, design project.	
PSoC 4100 BI	LE / PSoC 4200 BLE Design	Creates a PSoC 4100 BLE / PSoC 4200 BLE, 32-bit ARM Cortex-M0, design project	
PRoC BLE Des	ign	Creates a PRoC BLE, 32-bit ARM Cortex-M0, design project.	-
PSoC 4100M	/ PSoC 4200M Design	Creates a PSoC 4100M / PSoC 4200M, 32-bit ARM Cortex-M0, design project.	
PSoC 5LP Des	ign	Creates a PSoC 5LP, 32-bit ARM Cortex-M3, design project.	
PSoC 3 Starter Design	15		
■ PSoC 4000 Starter De	signs		
PSoC 4100 / PSoC 42	200 Starter Designs		
PSoC 4100 BLE / PSo	C 4200 BLE Starter Designs		+
Name:	Design01		
Location:	C:\Users\srds\Documents		
Device:	CY8C5868AXI-LP035	▼	
Workspace:	Create New Workspace	v]	
Workspace name:	Design01		
Project template:	Empty schematic	▼	
		OK	el

Figure 6-45. Opening New Project in PSoC Creator

- 2. Develop your custom project.
- 3. Build the project in PSoC Creator by selecting Build > Build Project or [Shift]+[F6].
- 4. Connect the 10-pin connector of MiniProg3 to the onboard 10-pin SWD debug and programming header J7 (which needs to be populated).
- To program the PSoC 5LP with PSoC Creator, click Debug > Program or [Ctrl]+[F5]. The Programming window shows MiniProg3 and the selected device in the project under it (CY8C5868LTI-LP039).
- 6. Click on the device and click Connect to program.



Notes:

- 1. The 10-pin SWD debug and programming header (J7) is not populated. See the 'No Load Components' section of A.6 Bill of Materials (BOM) for details.
- The PSoC 5LP pins are brought to the PSoC 5LP GPIO header (J8). These pins are selected to support high-performance analog and digital projects. See A.2 Pin Assignment Table for pin information.
- Take care when allocating the PSoC 5LP pins for custom applications. For example, P2[0]–P2[4] are dedicated for programming the PSoC 4. Refer to A.1 CY8CKIT-042 Schematics before allocating the pins.
- 4. When a normal project is programmed onto the PSoC 5LP, the initial capability of the PSoC 5LP to act as a programmer, USB-UART bridge, or USB-I2C bridge in not available.
- 5. The status LED does not function unless used by the custom project.

6.4 **PSoC 5LP Factory Program Restore Instructions**

The CY8CKIT-042 PSoC 4 Pioneer Kit features a PSoC 5LP device that comes factory-programmed as the onboard programmer and debugger for the PSoC 4 device.

In addition to creating applications for the PSoC 4 device, you can also create custom applications for the PSoC 5LP device on this kit. For details, see section 6.3 Developing Applications for PSoC 5LP on page 88. Reprogramming or bootloading the PSoC 5LP device with a new flash image will overwrite the factory program and forfeit the ability to use the PSoC 5LP device as a programmer/ debugger for the PSoC 4 device. Follow the instructions to restore the factory program on the PSoC 5LP and enable the programmer/debugger functionality.

6.4.1 PSoC 5LP is Programmed with a Bootloadable Application

If the PSoC 5LP is programmed with a bootloadable application, restore the factory program by using one of the following two methods.

6.4.1.1 Restore PSoC 5LP Factory Program Using PSoC Programmer

- 1. Launch **PSoC Programmer 3.23.1** or later from **Start > Cypress > PSoC Programmer**.
- Configure the Pioneer Kit in Service Mode. To do this, while holding down the reset button (SW1 Reset), plug in the PSoC 4 Pioneer Kit to the computer using the included USB cable (USB A to mini-B). This puts the PSoC 5LP into service mode, which is indicated by the blinking green status LED.



3. The following message appears in the PSoC Programmer results window "KitProg Bootloader device is detected".

Figure 6-46. PSoC Programmer Results Window

PSoC Programmer	
File View Options Help	
🚰 - 🔪 💿 BB 🚺	
Port Selection Progra	ammer Utilities JTAG
Prog	gramming Parameters
File	Path: C:\Program Files (x86)\Cypress\CY8CKIT-042 PSoC 4 Pioneer Kit\1.0\Firmware\Programmer\KitPr
Proc	grammer:
Proc	gramming Mode: Reset Power Cycle Power Detect
	ification: On Off <u>Connector:</u> 5p 10p
Device Family	Detection: On Off <u>Clock Speed:</u> 1.6 MHz
	grammer Characteristics Status Execution Time:
Device	age: 50V 33V 25V 18V
CY8C5868LTI-LP039 -	<u>Voltage:</u> NA
Actions	Results
Connected at 6:40:11 PM	KitProg bootloader device is detected
	Please close all ports, then navigate to the Utilities tab and click the Upgrade Firmware button to recover Bridge
	Select Port in the PortList, then try to connect
Device set to CY8C5868LTI-LP039 at	
6:38:29 PM	262144 FLASH bytes
Device Family set to	
CY8C5xxxLP at 6:38:29 PM	
Active HFX file set at	C.\Program Files (x86)\Cypress\CY8CKIT-042 PSoC 4 Pioneer
Active HEX file set at 6:38:28 PM	C:\Program Files (x86)\Cypress\CY8CKIT-042 PSoC 4 Pioneer Kit\1.0\Firmware\Programmer\KitProg\KitProg.hex
	Kit\1.0\Firmware\Programmer\KitProg\KitProg.hex
	Kit\1.0\Firmware\Programmer\KitProg\KitProg.hex Users must be aware that the following PSoC device should not be powered or

 Switch to the Utilities tab in PSoC Programmer and press the Upgrade Firmware button. Unplug all other PSoC programmers (such as MiniProg3 and DVKProg) from the PC before pressing the Upgrade Firmware button.

Figure 6-47. Upgrade Firmware

PSoC Programmer			3
File View Options Help			
🔁 · 🔪 💿 BB 🖉			
Port Selection Progr	ammer Utilities JTAG		
Device Family CY8C5xxLP Device CY8C5868LTI-LP039	Upgrade Firmware Click to upgrade connected device's firmware Erase Block Click to erase user specific flash block		
Actions	Results	-	-
Connected at 6:40:11 PM	KitProg bootloader device is detected Please close all ports, then navigate to the Utilities tab Upgrade Firmware button to recover Bridge) and click the	



5. After programming has completed, the following message appears: "Firmware Update Finished at <time>".

Figure 6-48. Firmware Update Complete

PSoC Programmer	
File View Options Help	
🖆 · 🗼 💿 🖻 🕻 🗎 🗋 🕒 🕲	
Port Selection Vilities JTAG	
KitProg/110E192D00232400 Upgrade Firmware Click to upgrade connected device's firmware Erase Block Click to erase user specific flash block	
Device Family	
CY8C5xoxLP v	
Device	
CY8C5868LTI-LP039 *	
Actions Results	
Successfully Connected to KitProg/110E192D00232400 at 6:42:32 PM Opening Port at 6:42:29 PM	
Connected at 6:42:29 PM KitProg/110E192D00232400	
Disconnected at 6:42:28 PM Bootloader device	≡.
Firmware Update Finished at 6:42:27 PM	
Succeeded	
Verifying	
Upgrading	
Initializing Firmware Upgrade Started at 6:42:21 PM	
Firmware Upgrade Started at 6:42:21 PM	
Connected at 6:40:11 PM KitProg bootloader device is d	letected
Flease close all ports, then n and click the Upgrade Firmware	avigate to the Utilities tab button to recover Bridge
Select Port in the PortList, t	hen try to connect 💡
For Help, press F1	PASS Powered Connected

6. The factory program is now successfully restored on the PSoC 5LP. It can be used as the programmer/debugger for the PSoC 4 device.



6.4.1.2 Restore PSoC 5LP Factory Program Using USB Host Tool

- 1. Launch the Bootloader Host tool from **Start > Cypress > PSoC Creator**.
- 2. Using the **File > Open** menu, load the *Kit Prog.cyacd* file, which is installed with the kit software. The default location for this file is: <Install_Directory>\CY8CKIT-042 PSoC 4 Pioneer Kit\<version>\Firmware\Programmer\KitProg\KitProg.cyacd

Bootloader Host	
Eile Actions Help	
File: C:\Program Files (x86)\Cypress\CY8CKIT-042 PSoC 4 Pioneer Kit\1.0\Firmware\Programmer\KitProg\KitProg.cyacd	
Ports: Filters Port Configuration UART Port Information Generic Serial Port	
Baud 9600 -	
Data Bits 8	
Stop Bits One	
Parity None	
Log: 06:15:14 PM - Selected device: Communications Port (COM1)	
06:15:14 PM - Selected device: Communications Port (COM1)	
Ready	.:

Figure 6-49. Load KitProg.cyacd File

🛓 Open						×
COO - 🥻 « Firmware	▶ Programmer ▶ k	GitProg 👻 😽	Search KitPi	rog		٩
Organize 🔻 New folder					-	0
🔆 Favorites	Name	Date modified	Туре	Size		
Nesktop	KitProg.cyacd	4/18/2013 1:07 AM	CYACD File		121 KB	
 Downloads Recent Places Google Drive Dropbox No-Zoolz Zone SkyDrive Libraries Apps Documents 						
	ne: KitProg.cyacd	•	Bootloader F	iles	Cance	•



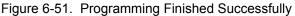
- Configure the Pioneer Kit in Service Mode. To do this, while holding down the reset button (SW1 Reset), plug in the PSoC 4 Pioneer Kit to the computer using the included USB cable (USB A to mini-B). This puts the PSoC 5LP into service mode, which is indicated by the blinking green status LED.
- 4. In the Bootloader Host tool, set the filters for the USB devices with VID: 04B4 and PID: F13B. **USB Human Interface Device** port appears in the Ports list. Click that port to select it.

Bootloader Host				
<u>File Actions H</u> elp				
🖆 💫 BB 🔇	\geqslant			
File: C:\Program Files (x8	6)\Cypress\CY8CKIT-042 PS	oC 4 Pioneer Kit\1.0\Firmware	\Programmer\KitProg\KitProg.cyacd	
Ports: Communications Port USB Human Interface USB Log:	(COM1) No co port.	Configuration USB onfiguration necessary for this	VID: 0484 PID: F138	
06:15:14 PM - Selected de 06:15:14 PM - Selected de 06:21:45 PM - Selected de 06:25:53 PM - Selected de 06:25:53 PM - Selected de 06:25:58 PM - Selected de	vice: Communications Port (C vice: USB Human Interface [vice: USB Human Interface [vice: Communications Port (C vice: USB Human Interface [vice: Communications Port (C vice: USB Human Interface [XOM1) Device (04B4_F13B) XOM1) Device (04B4_F13B) XOM1)		
Ready				

Figure 6-50. Select USB Human Interface Device

- 5. Click the **Program** button (or menu item **Actions > Program**) to restore the factory-program by bootloading it onto the PSoC 5LP.
- 6. After programming has completed, the following message appears: "Programming Finished Successfully".





Bootloader Host	
Eile Actions Help	
File: C:\Program Files (x86)\Cypress\CY8CKIT-042 PSoC 4 Pioneer Kit\1.0\Firmware\Programmer\KitProg\KitProg.cyacd	
Potts: Filters Pott Configuration UART Pott Information Communications Port (COM1) Baud 9600 Image: Second Port Second Po	
Log: 06:15:14 PM - Selected device: Communications Port (COM1)	
06:15:14 PM - Selected device: Communications Port (COM1) 06:15:14 PM - Selected device: Communications Port (COM1) 06:25:14 PM - Selected device: Communications Port (COM1) 06:25:53 PM - Selected device: Communications Port (COM1) 06:25:53 PM - Selected device: Communications Port (COM1) 06:25:53 PM - Selected device: USB Human Interface Device (0484_F138) 06:26:53 PM - Selected device: USB Human Interface Device (0484_F138) 06:26:54 PM - Selected device: USB Human Interface Device (0484_F138) 06:26:34 PM - Selected device: USB Human Interface Device (0484_F138) 06:29:34 PM - Programming Stated 06:29:34 PM - Programming Stated 06:29:34 PM - Selected device: Communications Port (COM1) 06:29:34 PM - Programming Stated 06:29:34 PM - Programming Finished Successfully Programming completed in 519 Ims. 06:29:34 PM - Selected device: Communications Port (COM1)	
Ready	

7. The factory program is now successfully restored on the PSoC 5LP. It can be used as the programmer/debugger for the PSoC 4 device.

6.4.2 PSoC 5LP is Programmed with a Standard Application

If PSoC 5LP is programmed with a standard application, restore the factory program by using the following method.

- 1. Launch **PSoC Programmer 3.23.1** or later from **Start > Cypress > PSoC Programmer**.
- 2. Use the File > Open menu to load the *KitProg.hex* factory program hex file, which is shipped with the kit. The default location for this file is: <Install_Directory>\CY8CKIT-042 PSoC 4 Pioneer Kit\<version>\Firmware\Programmer\KitProg
- Connect a CY8CKIT-002 MiniProg3 (sold separately) to the computer. The 10-pin connector cable on the MiniProg3 plugs into the header [J7]. Note that the J7 header is unpopulated. For more details, see A.6 Bill of Materials (BOM) on page 121.



4. Ensure that **MiniProg3** is the selected port in PSoC Programmer and the 10-pin connector (**10p** option) is selected, as shown in the following figure. If the board is not powered over USB, select the **Power Cycle** programming mode.

Figure 6-52. Select MiniProg3

File View Options Help Image: State Stat
Port Selection Image: Constraint of the constraint of th
MiniProg3/3209AA00002 Programming Parameters File Path: C:\Program Files (x86))Cypress/CY8CKIT-042 PSoC 4 Pioneer Kit1.0/FirmwarelProgrammer/KitProg/KitProg.hex Programming Mode: Reset Power Detect Verification: On Off Connector: Sp 10 Programming Characteristics Device Family Programmer: Programmer: Device Family On Off Clock Speed:
File Path: C:Program Files (x86)/Cypress/CY8CKIT-042 PSoC 4 Pioneer Kit1 0/FirmwarelProgrammer/KitProg/KitProg hex Programmer: Mini/Prog3/3209AA000002 Programming Mode: © Reset © Power Detect Verification: © On © Off Clock Speed: 16 MHz • Programmer Characteristic Status
Programming Mode: MiniProg3/3209AA000002 Programming Mode: Reset Power Detect Verification: On Off Connector: 5p 10 MiniProg3/3209AA000002 Programming Mode: Status Status Device Family Programmer Characteristics Status
Programming Mode: © Reset © Power Detect Verification: © On Off Connector: © 5p © 10p Device Family AutoDetection: © On Off Clock Speed: 16 MHz •
Programming Mode: © Reset @ Power Oycle Power Detect Verification: @ On Off Connector: 5p @ 10p Device Family AutoDetection: @ On Off Clock Speed: 16 MHz •
Device Family Device Family Programmer Characteristics Programmer Characteristics Status St
Device Family Programmer Characteristics Status
Programmer Characteristice
Device Protocol: JTAG @ SWD @ ISSP @ I2C Device Voltage: 0 SWD @ ISSP @ I2C
Uevee Uevee 0 5.0 V 3.3 V 2.5 V 1.8 V Urgersessurge
Actions Results
MiniProg3/3209AA000002 at 6:58:42 PM MiniProg3 Version 2.05 [3.08/2.05]
Opening Port at 6:58:41 EM
For Help, press F1 FAIL Not Powered Connected

5. When ready, press the **Program** button (or **File > Program**) to program the PSoC 5LP device.



6. After programming has completed, the following message appears: "Program Finished at <time>".

Figure 6-53. Program Finished

PSoC Programmer		- • ×
File View Options Help		
Pot Selection I Utilities JTAG		
MiniProg33209A4000002 Programming Parameters		
File Path: C:\Program Files (x88)/Cypress\CY8CKIT-042 PSoC 4 Pioneer Kith1.0/Firmware\Programmer/KitProg.hex		
Program Button Programmer, MiniProg3/3209AA000002		
Programming Mode: Reset Power Cycle Power Detect		
Verification: On Off <u>Connector</u> 5 p 10p		
Device Family Over Family Over Clock Speed: 1.6 MHz V		
CY8C5collP - Programmer Characteristics Status Execution Time: 19.5 seconds		
Device State State State Off		
<u>Voltage:</u> ● 5.0 ∨ ● 3.3 ∨ ● 2.5 ∨ ● 1.8 ∨ <u>Voltage:</u> ● 60 mV		
Actions Results		
Program Finished at 7:00:04 PM		
Programming Succeeded		
Doing Checksum		
Doing Protect		
Programming of Flash Succeeded		
Programming of Flash Starting		
Erase Succeeded Device set to CY8C5868LTI-LF039 at 6:59:55		
PM PM		
Device Family set to CY8C5xxxLP at 6:59:55		
FM Automatically Detected Device: CY8C5868LTI-LP039		
Program Requested at 6:59:44 PM		
Successfully Connected to MiniProg3 version 2.05 [3.08/2.05]		
MiniProg3/3209AA000002 at 6:58:42 PM		
Opening Fort at 6:58:41 PM		
For Help, press F1 PASS Not Pow	ered 🛛	Connected:

7. The factory program is now successfully restored on the PSoC 5LP. It can be used as the programmer/debugger for the PSoC 4 device.

6.5 Using µC/Probe Tool

Micrium's µc/probe is a windows application that allows you to read and write the memory of any embedded target processor during run-time, and map those values to a set of virtual controls and indicators placed on a graphical dashboard.

This tool helps in designing a simple GUI for the example projects of the development kits with least effort.

Please note that Micrium μ C/Probe tool is not pre-requisite software required to run this kit and does not get installed along with kit contents.

The license required to use all the features of the tool need to be purchased separately by the user. However, the educational edition of the tool is available as a free download from http://micrium.com/download/ucprobe-3-0-trial-version/.

The Educational Edition of μ C/Probe is available for free to enable you to "try before you buy". For more details on licensing and the μ C/Probe tool, please refer to the μ C/Probe Users' Manual at http://micrium.com/download/%c2%b5cprobe-3-0-users-manual/.

In Micrium μ C/Probe version 3.3, the Cypress KitProg is being supported as a means of communication to the target device connected to PC.



When an example project is built in PSoC Creator, it produces the output files in HEX, LST, MAP, RPT and ELF formats.

The ELF file lists all the symbols (variables), symbol types and its addresses. The Micrium μ C/Probe tool reads the ELF file and detects these symbols (global variables) used in the code.

The μ C/Probe tool provides a host of graphical controls such as sliders, RGB palette, graphs, donuts etc. The controls required can be dragged and dropped onto μ C/Probe workbench and symbols from ELF file can be assigned to the controls.

When the workbench is run, the changes in symbol value associated with the controls (slider, graph etc) can be visualized on the PC.

Appropriate global variables have been assigned in CapSense and PWM example projects of the kit to visualize the CapSense output and PWM output on a GUI.

Note: The ELF file is generated by the PSoC Creator and gets removed during the project Clean process. So, to use any of the WSPX files, please ensure to build the project so that ELF file gets generated. Else, the μ C/Probe tool pops-up a message the ELF file is missing.

6.5.1 CapSense Example Project

To visualize the output of CapSense project using μ C/Probe tool, follow the steps given below:

- 1. Program the CapSense example project on CY8CKIT-042 by following steps 1–8 in chapter 5.
- 2. Download and install µC/Probe tool from http://micrium.com/download/µcprobe-3-0-trial-version/.
- 3. Launch µC/Probe from Start > All Programs > Micrium > uC-Probe > Micruim uC-Probe.

Figure 6-54. Micrium µC/Probe

960	Brand Star Strangenet Mark Street Street	kus0 🗰
	trafford - Xing - Etraffaite	Mionium
	tring Farward - Kalang - El Barrel Han Han and Januard - P auge - El Dennitem Hann	Micriµm
Annual Connect	L+OB them + Line/ter	Educational Edition
Writelite Controls		AT N R R H
1		
		a Mittelevet
-R Lase (as		
CBB California		
R and		
HE balance		
0.022		
BB Tage Life		
BH inset Liter		
The large		
S unright		
-		
CE tetes		
Without Correct		
Line Impe		
Persona Linee Sauger		
Generatings	La contra	
Territor Territ	der der der der der Anne (der seiter)	
Courdiage	tere Restored I for	Da barlinar Hereshilar
tellerat.		
Concesso .		
049		
Names Instation		
104		
Adapted		
,		
Depend	Plat Devin	

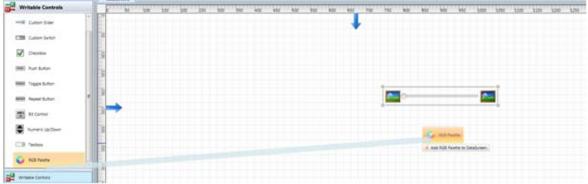


4. Drag and drop a Custom Slider control from Writable Controls in Toolbox on to the Datascreen1. Figure 6-55. Adding Slider Control

W	ritable Controls		123	10.00	11640	644	14224	1.644	1.640	his 1	1.000	1.66	1.000	14420	100	850 70		201134	11.444	1.44	11 2441	1.000	1 1.445	1100	1110	1244
*		+	P			1.00		- 191					-				×	+or	410			1000	1000	1000	- Sarre	
-	Cator Sider		5																							
	Culton Seltch		E																							
7	Owner		10																							
-	Fuel Satter		18																							
-	Topgie Butter		1																							
-	Nonet Button		-	*												+ AND Curt	er Date 1	a Detafired	a.)							
Ť.	B1 Cartrol		1																							
٥	Numeric Lip/Down		100																							
-	Teribix		-2																							
0	RGE Faurte		-																							
-	Mile Corthon		1																							

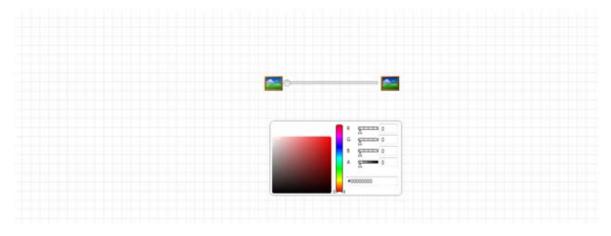
5. Next, add a RGB Palette from Writable Controls in Toolbox on to the Datascreen1.

Figure 6-56.	Adding RGB Palette Control
Torise	+ # Datalaseri



6. The DataScreen1 looks as below after adding both the controls.

Figure 6-57. DataScreen with Slider and RGB controls





7. Now, click on the ELF button in the Symbol Browser window.

Figure 6-58. ELF button in Symbol Browser

mbol Browser								
🖁 ELF 📑 COF	CSF 🚰 MQTT 🔎		Search by	y Name 🔘 Search by Da	ita Type 😻 🖄	RAM Range Min 0		Max FFFFFFF
		Name		Display		Name	Ð	

PSoC Device	
	PSoC Device

- 8. Browse and point to the CapSense.ELF file to load the symbols (global variables) from the CapSense example project. Wait until the ELF file gets loaded. The elf file is in the collapsed state by default. It can expanded by clicking on the '+' button next to the file name.
- 9. On expanding the elf file, it can be seen that the Symbol Browser displays all the .C files in the example project. By expanding each file, the global variables defined in that C file are displayed with its name, type, size, memory address etc.
- 10.Now, expand the main.c file to view the global variables defined in main.c file.

Figure 6-59. Global variables in Symbol Browser

Name		Display Name	4	Type	50
X Defenset	1	ua.		NA	0
iii Capteronic		Capterne.c		N/A	34
iii CapSerse,CDHLc		CapServie, CSHLic		NIA	4
ill Capterne_SMS.c	0	Desterne_SMS.c		NUA	
iii Cruttante	4	Inditants		N/A	
H QGB4		\$UB4		N/A	1
ili gmetadatac		ymetadata-c		NIA	
ii main.c		nánc		N/A	1
lamPosition		adPosition		short unsigned int	1
siderfactor		riderRosition		short unsigned int	1
ucARG8		eARG8		long unsigned int.	
idliter		Giter		unsigned char	3
iii PWMc		NWME		NUA	1



11. Drag and drop the global variable µcSlider on to the custom slider control to see the slider output. Similarly, drag and drop the global variable µcARGB on to the RGB Palette to see the RGB output.

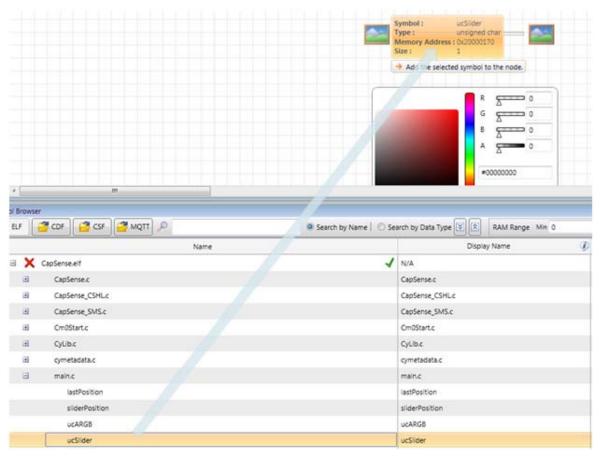


Figure 6-60. Assigning slider output to Custom Slider control



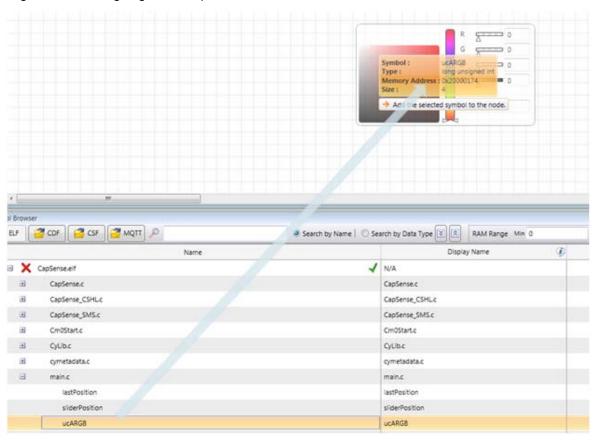
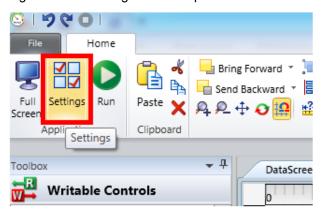


Figure 6-61. Assigning RGB output to RGB Palette control

12.Now, connect the CY8CKIT-042 to PC. Click on the Settings button in the μ C/Probe tool. Figure 6-62. Settings button in μ C/Probe





13. In the μ C/Probe setting window, select the Cypress PSoC Prog and select 'KitProg/<Kit Prog number>' from the drop down box for Port and click **OK** to start communication between the CY8CKIT-042 and the μ C/Probe tool.

Micriµm µC/Probe -23 µC/Probe Settings Endianness Statistics Data Collection Little Endian symbols/sec ı. I. ı . ı. I. I I Big Endian Obytes/sec Slowest Fastest C Interfaces Refresh Communication None Settings KitProg/191A07A0032C2400 Cloud Only Port KitProg/191A07A0032C2400 Target Resident Code M TCP/IP 🏄 RS-232 💰 USB 4 Debug Interfaces General 💰 J-Link 🔬 CMSIS-DAP Cypress PSoC Prog OK Cancel

Figure 6-63. µC/Probe Settings



14. Click on Run button to start.

Figure 6-64. Run button



- 15.Now, move your finger on the CapSense slider on the kit and observe the Custom Slider and RGB Palette control output on the datascreen.

Figure 6-65. Custom Slider and RGB Palette output

Note: If you are using Education edition of μ C/Probe tool, pop-up windows will be displayed before starting datascreen. Click OK to continue. Also, the Datascreen (output) will time-out after 1 minute in case of Education edition.

16.Click on File tab and select Save to save the μC/Probe project. Provide an appropriate name and select a location to save your project. The μC/Probe projects are saved with extension .WSPX. Double clicking a .WSPX file opens the μC/Probe tool.

6.5.2 PWM Example project

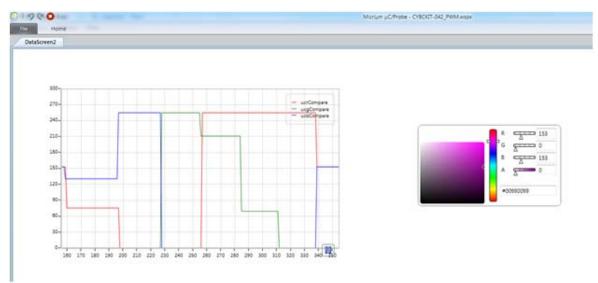
The μ C/Probe project for PWM project is already created and packaged along with kit content. The μ C/Probe projects can be found in the installation folder at <Install_Directory>\CY8CKIT-042 PSoC 4 Pioneer Kit\<version>\ μ CProbe.

- 1. Program the PWM example project on CY8CKIT-042 by following steps 1–8 in Chapter 5.
- 2. Double click on CY8CKIT-042_PWM.wspx file.



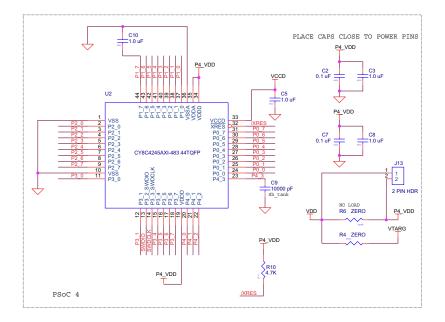
- 3. Browse and point to the PWM.elf file to load the symbols (global variables) from the PWM code example.
- 4. Connect the CY8CKIT-042 to PC and follow steps 12 to 14 described above to start running the datascreen.
- 5. The PWM compare values are displayed graphically and the RGB palette displays the RGB LED output on the datascreen.

Figure 6-66. PWM Compare values and RGB output

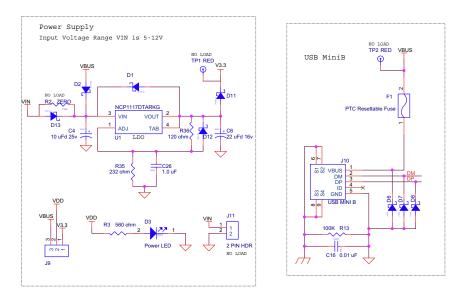


A. Appendix

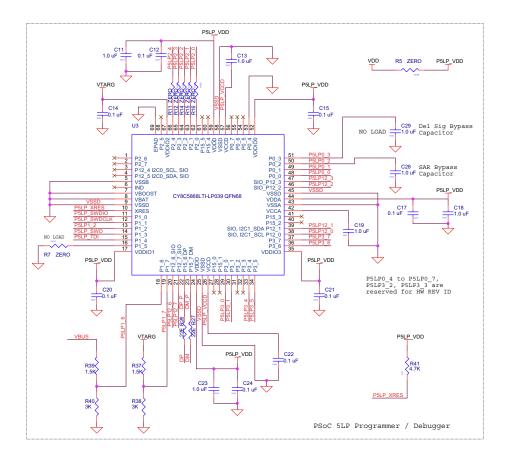


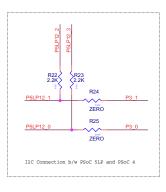


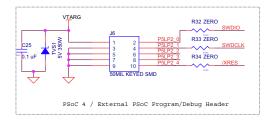
A.1 CY8CKIT-042 Schematics

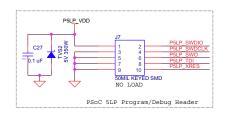




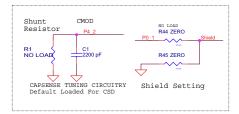


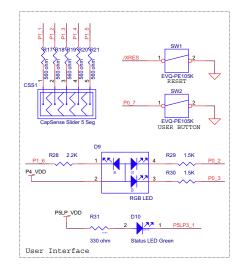


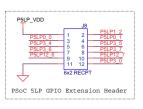




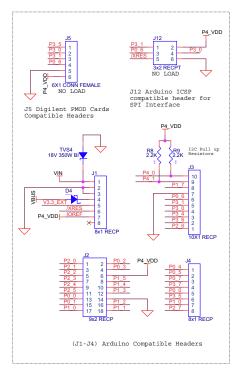




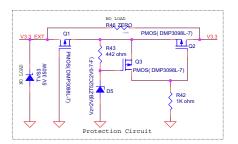












A.2 Pin Assignment Table

This section provides the pin map of the headers and their usage.

	J1					
Pin	Kit Signal	Description				
J1_01	VIN	Input voltage to the board				
J1_02	GND	GND				
J1_03	GND	GND				
J1_04	5V	5 V voltage				
J1_05	3.3V	3.3 V voltage				
J1_06	RESET	/XRES				
J1_07	IOREF	I/O voltage reference				
J1_08	NC	Not connected				

A.2.1 Arduino Compatible Headers (J1, J2, J3, J4, and J12)

	J2							
Pin	PSoC 4 Signal	PSoC 4 Description	Pin	PSoC 4 Signal	PSoC 4 Description			
J2_01	P2[0]	A0 (SARADC input)	J2_02	P0[2]	Comparator 2+			
J2_03	P2[1]	A1 (SARADC input)	J2_04	P0[3]	Comparator 2–			
J2_05	P2[2]	A2 (SARADC input)	J2_06	VDD	VDD			
J2_07	P2[3]	A3 (SARADC input)	J2_08	P1[5]	Opamp 2+			
J2_09	P2[4]	A4 (SARADC input)	J2_10	P1[4]	Opamp 2–			
J2_11	P2[5]	A5 (SARADC input)	J2_12	P1[3]	Opamp 2out			
J2_13	P0[0]	Comparator 1+	J2_14	GND	GND			
J2_15	P0[1]	Comparator 1–	J2_16	P1[2]	Opamp 1out			
J2_17	P1[0]	Opamp 1+	J2_18	P1[1]	Opamp 1–			



	J3					
Pin	PSoC 4 Signal	PSoC 4 Description				
J3_01	P2[6]	D8				
J3_02	P3[6]	D9(PWM)				
J3_03	P3[4]	D10(PWM/SS)				
J3_04	P3[0]	D11(PWM/MOSI)				
J3_05	P3[1]	D12(MISO)				
J3_06	P0[6]	D13(SCK)				
J3_07	GND	GND				
J3_08	P1[7]	AREF				
J3_09	P4[1]	SDA				
J3_10	P4[0]	SCL				

J4						
Pin	PSoC 4 Signal	PSoC 4 Description				
J4_01	P0[4]	D0(RX)				
J4_02	P0[5]	D1(TX)				
J4_03	P0[7]	D2				
J4_04	P3[7]	D3(PWM)				
J4_05	P0[0]	D4				
J4_06	P3[5]	D5(PWM)				
J4_07	P1[0]	D6(PWM)				
J4_08	P2[7]	D7				

J12							
Pin	Kit Signal	PSoC 4 Description					
J12_01	P3[1]	MISO					
J12_02	PSoC 4_VDD	VDD					
J12_03	P0[6]	SCK					
J12_04	P3[0]	MOSI					
J12_05	/XRES	PSoC 4 RESET					
J12_06	GND	GND					



A.2.2 Digilent Pmod Cards Support Header (J5)

J5						
Pin Kit Signal		PSoC 4 Description (Default Pmod Signals)				
J5_01	P3[5]	SPI_SS (multiplex with J4_06)				
J5_02	P3[0]	SPI_MOSI				
J5_03	P3[1]	SPI_MISO				
J5_04	P0[6]	SPI_SCK				
J5_05	GND	GND				
J5_06	VDD	VCC				



A.2.3 PSoC 5LP GPIO Header (J8)

J8 is a 2×6 header that connects PSoC 5LP pins to support GPIO controls for custom PSoC 5LP projects.

	J8								
Pin	Pin PSoC 5LP Signal PSoC 5LP Des		Pin	PSoC 5LP Signal	PSoC 5LP Description				
J8_01	PSoC 5LP_VDD	VDD	J8_02	P1[2]	Digital I/O				
J8_03	P0[0]	Delta Sigma ADC + input	J8_04	P0[1]	Delta Sigma ADC – input				
J8_05	P3[4]	SAR – input	J8_06	P3[5]	SAR + input				
J8_07	P3[6]	Buffered VDAC	J8_08	P3[7]	Buffered VDAC				
J8_09	P12[6]	UART RX	J8_10	P12[7]	UART TX				
J8_11	GND	GND	J8_12	P3[0]	IDAC output				

A.3 Program and Debug Headers

A.3.1 PSoC 4 Direct Program/Debug Header (J6)

	J6								
Pin	PSoC 5LP Signal	PSoC 4 Signal	Description	Pin	PSoC 5LP Signal	PSoC 4 Signal	Description		
J6_01	VDD	VDD	VCC	J6_02	P2[0]	P3[2]	TMS/SWDIO		
J6_03	GND	GND	GND	J6_04	P2[1]	P3[3]	TCLK/SWCLK		
J6_05	GND	GND	GND	J6_06	P2[2]	NC	TDO/SWO		
J6_07	NC	GND	GND	J6_08	P2[3]	NC	TDI		
J6_09	GND	GND	GND	J6_10	P2[4]	XRES	RESET		

A.3.2 PSoC 5LP Direct Program/Debug Header (J7)

	J7								
Pin	PSoC 5LP Signal	Description	Pin	PSoC 5LP Signal	Description				
J7_01	VDD	VCC	J7_02	P1[0]	TMS/SWDIO				
J7_03	GND	GND	J7_04	P1[1]	TCLK/SWCLK				
J7_05	GND	GND	J7_06	P1[3]	TDO/SWO				
J7_07	GND	GND	J7_08	P1[4]	TDI				
J7_09	GND	GND	J7_10	XRES	RESET				



A.4 Use of Zero-ohm Resistors and No Load

Unit	Resistor	Usage
Power supply	R2	Solder zero-ohm resistors to access voltage from VBUS (USB).
I2C connection between PSoC 5LP and PSoC 4	R24 and R25	Unsolder the resistors to communicate with an external PSoC using the PSoC 5LP. Removing these will disable the PSoC 4 I2C communication with the PSoC 5LP device.
PSoC 4/external PSoC program/ debug header	R32, R33, and R34	Unsolder the resistors to disconnect SWD lines from the PSoC 4. Use J6 to connect and program an external PSoC. Removing these will disable PSoC 4 programming by the PSoC 5LP device.
Protection circuit	R46	Solder zero-ohm resistors to bypass the entire protection circuitry.
CapSense tuning circuitry	R1	Used when RBleed mode of the CSD is used. To use this feature, you must populate an Rbleed resistor. Refer to the CapSense component datasheet.
CapSense shield setting	R44, R45	Unsolder R45, which connects the shield to ground and solder R44 with zero-ohm resistors to connect Vref via P0_1.
PSoC 4	R4, R6	Unsolder R4 to remove supply to VTARG and solder zero-ohm resistors R6 to supply P4_VDD with VDD instead of J13.
PSoC 5LP programmer/debugger	R11, R12, R14, R15, R16	For future use.
	R5	Unsolder the zero-ohm resistor to cut the VDD supply to PSoC 5LP.
	R7	For future use.

A.5 Error in Firmware/Status Indication in Status LED

	User Indication	Scenario	Action Required by user
1	LED blinks at a fast rate (ON Time = 0.25s, OFF Time = 0.25s)	Bootloadable file is corrupt	Bootload the *.cyacd file over the USB interface, which is shipped with PSoC Programmer using the Bootloader Host GUI shipped with PSoC Creator. The files are located in the PSoC Programmer root instal- lation directory.
2	LED blinks at a slow rate (ON Time = 1.5s, OFF Time = 1.5s)	Entered Boot- loader by press- ing the PSoC 4 Reset switch	 a) Unplug power and plug it in again if you entered this mode by mistake; the LED gives the indication. b) If the mode entry was intentional, bootload the new *.cyacd file using the Bootloader Host tool shipped with PSoC Creator.
3	LED glows steadily	Programmer appli- cation is running successfully	USB is enumerated successfully and the programmer is up and running.The PSoC 4 device can now be pro- grammed any time using the onboard PSoC 5LP pro- grammer.

Note: LED status is not applicable when a custom project is running in PSoC 5LP.



A.6 Bill of Materials (BOM)

No.	Qty	Reference	Value	Description	Manufacturer	Mfr Part Number
1				PCB,3.32"x2.1" CAF resistant High Tg ENIG finish, 4 layer, Color = RED, Silk = WHITE.	Cypress	
2	1	C1	2200 pFd	CAP CER 2200PF 50V 5% NP0 0805	Murata	GRM2165C1H222JA0 1D
3	12	C2,C7,C12,C14,C15,C 17,C20,C21,C22,C24, C25,C27	0.1 uFd	CAP .1UF 16V CERAMIC Y5V 0402	Panasonic - ECG	ECJ-0EF1C104Z
4	11	C3,C5,C8,C10,C11,C1 3,C18,C19,C23,C26,C 28	1.0 uFd	CAP CERAMIC 1.0UF 25V X5R 0603 10%	Taiyo Yuden	ТМК107ВЈ105КА-Т
5	1	C4	10 uF 25V	CAP TANT 10UF 25V 10% 1210	AVX Corporation	TPSB106K025R1800
6	1	C6	22 uF 16V	CAP TANT 22UF 16V 10% 1210	AVX Corporation	TPSB226K016R0600
7	1	C9	10000 pFd	CAP CER 10000PF 50V 5% NP0 0805	Murata	GRM2195C1H103JA0 1D
8	1	C16	0.01 uFd	CAP 10000PF 16V CERAMIC 0402 SMD	Panasonic - ECG	ECJ-0EB1C103K
9	6	D1,D2,D4,D11,D12,D1 3	MBR05	DIODE SCHOTTKY 0.5A 20V SOD- 123	Fairchild Semicon- ductor	MBR0520L
10	1	D3	Power LED Amber	LED AMBER 591NM DIFF LENS 2012	Sharp Microelectron- ics	LT1ZV40A
11	1	D5	2V Zener	DIODE ZENER 2V 500MW SOD123	Diodes Inc	BZT52C2V0-7-F
12	3	D6, D7, D8	ESD diode	SUPPRESSOR ESD 5VDC 0603 SMD	Bourns Inc.	CG0603MLC-05LE
13	1	D9	RGB LED	LED RED/GREEN/BLUE PLCC4 SMD	Cree, Inc.	CLV1A-FKB- CJ1M1F1BB7R4S3
14	1	D10	Status LED Green	LED GREEN CLEAR 0805 SMD	Chicago Miniature	CMD17-21VGC/TR8
15	1	F1	FUSE	PTC Resettable Fuses 15Volts 100Amps	Bourns	MF-MSMF050-2
16	2	J1, J4	8x1 RECP	CONN HEADER FEMALE 8POS .1" GOLD	Sullins Connector Solutions	PPPC081LFBN-RC
17	1	J2	9x2 RECP	CONN HEADER FMAL 18PS.1" DL GOLD	Sullins Connector Solutions	PPPC092LFBN-RC
18	1	J3	10x1 RECP	CONN HEADER FMALE 10POS .1" GOLD	Sullins Connector Solutions	PPPC101LFBN-RC
19	1	J6	50MIL KEYED SMD	CONN HEADER 10 PIN 50MIL KEYED SMD	Samtec	FTSH-105-01-L-DV-K
20	1	J8	6x2 RECP	CONN HEADER FMAL 12PS.1" DL GOLD	Sullins Connector Solutions	PPPC062LFBN-RC
21	1	J9	3p_jumper	CONN HEADER VERT SGL 3POS GOLD	ЗМ	961103-6404-AR
22	1	J10	USB Mini B	CONN USB MINI AB SMT RIGHT ANGLE	TE Connectivity	1734035-2



No.	Qty	Reference	Value	Description	Manufacturer	Mfr Part Number
23	1	J13	2p_jumper	CONN HEADER VERT SGL 2POS GOLD	ЗМ	961102-6404-AR
24	3	Q1,Q2,Q3	PMOS	MOSFET P-CH 30V 3.8A SOT23-3	Diodes Inc	DMP3098L-7
25	1	R3	560 Ω	RES 560 Ω 1/8W 5% 0805 SMD	Panasonic - ECG	ERJ-6GEYJ561V
26	12	R4,R11,R12,R14,R15, R16,R24,R25,R32,R33 ,R34,R45	ZERO	RES 0.0 Ω 1/10W 0603 SMD	Panasonic-ECG	ERJ-3GEY0R00V
27	1	R5	ZERO	RES 0.0 Ω 1/8W 0805 SMD	Panasonic-ECG	ERJ-6GEY0R00V
28	4	R8,R9,R22,R23	2.2K	RES 2.2 kΩ 1/10W 5% 0603 SMD	Panasonic - ECG	ERJ-3GEYJ222V
29	2	R10,R41	4.7K	RES 4.7 kΩ 1/10W 5% 0603 SMD	Panasonic-ECG	ERJ-3GEYJ472V
30	1	R13	100K	RES 100 kΩ 1/10W 5% 0402 SMD	Panasonic - ECG	ERJ-2GEJ104X
31	5	R17,R18,R19,R20,R21	560 Ω	RES 560 Ω 1/10W 5% 0603 SMD	Panasonic-ECG	ERJ-3GEYJ561V
32	2	R26, R27	22E	RES 22 Ω 1/10W 1% 0603 SMD	Panasonic - ECG	ERJ-3EKF22R0V
33	1	R28	2.2K	RES 2.2 kΩ 1/8W 5% 0805 SMD	Panasonic - ECG	ERJ-6GEYJ222V
34	2	R29,R30	1.5K	RES 1.5 kΩ 1/8W 5% 0805 SMD	Panasonic - ECG	ERJ-6GEYJ152V
35	1	R31	330 Ω	RES 330 Ω 1/8W 5% 0805 SMD	Panasonic - ECG	ERJ-6GEYJ331V
36	1	R35	232 Ω	RES 232 Ω 1/10W 1% 0603 SMD	Panasonic - ECG	ERJ-3EKF2320V
37	1	R36	120 Ω	RES 120 Ω 1/10W 1% 0603 SMD	Panasonic - ECG	ERJ-3EKF1200V
38	2	R37,R39	1.5K	RES 1.5K Ω 1/10W 5% 0603 SMD	Panasonic - ECG	ERJ-3GEYJ152V
39	2	R38,R40	зк	RES 3.0K Ω 1/10W 5% 0603 SMD	Panasonic - ECG	ERJ-3GEYJ302V
40	1	R42	1K	RES 1K Ω 1/8W 5% 0805 SMD	Panasonic - ECG	ERJ-6GEYJ102V
41	1	R43	442 Ω	RES 442 Ω 1/10W 1% 0603 SMD	Panasonic - ECG	ERJ-3EKF4420V
42	2	SW1,SW2	SW PUSH- BUTTON	SWITCH TACTILE SPST-NO 0.05A 12V	Panasonic - ECG	EVQ-PE105K
43	1	TP5	BLACK	TEST POINT PC MINI .040"D Black	Keystone Electronics	5001
44	2	TVS1,TVS2	5V 350W	TVS UNIDIR 350W 5V SOD-323	Dioded Inc.	SD05-7
45	1	TVS4	18V 350W	TVS DIODE 18V 1CH BI SMD	Bourns Inc.	CDSOD323-T18C
46	1	U1	NCP1117DT ARKG	NCP1117DTARKG	ON Semiconductor	NCP1117DTARKG
47	1	U2	PSoC 4 (CY8C4245A XI-483)	44TQFP PSoC4A target chip	Cypress Semicon- ductor	CY8C4245AXI-483
48	1	U3	PSoC 5LP (CY8C5868L TI-LP039)	68QFN PSoC 5LP chip for USB debug channel and USB-Serial interface	Cypress Semicon- ductor	CY8C5868LTI-LP039
No L	oad C	omponents			•	
49	1	C29	1.0 uFd	CAP CERAMIC 1.0UF 25V X5R 0603 10%	Taiyo Yuden	TMK107BJ105KA-T



No.	Qty	Reference	Value	Description	Manufacturer	Mfr Part Number
50	1	J5	6X1 RECP RA	CONN FEMALE 6POS .100" R/A GOLD	Sullins Connector Solutions	PPPC061LGBN-RC
51	1	J7	50MIL KEYED SMD	CONN HEADER 10 PIN 50MIL KEYED SMD	Samtec	FTSH-105-01-L-DV-K
52	1	J11	2 PIN HDR	CONN HEADER FEMALE 2POS .1" GOLD	Sullins Connector Solutions	PPPC021LFBN-RC
53	1	J12	3x2 RECPT	CONN HEADER FMAL 6PS .1" DL GOLD	Sullins Connector Solutions	PPPC032LFBN-RC
54	5	R1,R2,R7,R44,R46	ZERO	RES 0.0 Ω 1/10W 0603 SMD	Panasonic-ECG	ERJ-3GEY0R00V
55	1	R6	ZERO	RES 0.0 Ω 1/8W 0805 SMD	Panasonic-ECG	ERJ-6GEY0R00V
56	2	TP1,TP2	RED	TEST POINT PC MINI .040"D RED	Keystone Electronics	5000
57	3	TP3,TP4,TP6	BLACK	TEST POINT PC MINI .040"D Black	Keystone Electronics	5001
58	1	TVS3	5V 350W	TVS UNIDIR 350W 5V SOD-323	Dioded Inc.	SD05-7
Insta	ll on E	Bottom of PCB As per t	he Silk Screen	in the Corners		
59	4	N/A	N/A	BUMPON CYLINDRICAL.312X.215 BLACK	3M	SJ61A6
Spec	ial Ju	mper Installation Instru	ctions			
60	1	J9	Install jumper across pins 1 and 2	Rectangular Connectors MINI JUMPER GF 6.0MM CLOSE TYPE BLACK	Kobiconn	151-8010-E
61	1	J13	Install jumper across pins 1 and 2	Rectangular Connectors MINI JUMPER GF 6.0MM CLOSE TYPE BLACK	Kobiconn	151-8010-E
Labe						<u> </u>
62	1	N/A	N/A	LBL, Kit Product Identification Label, Vendor Code, Datecode, Serial Number CY8CKIT-042 Rev** (YYWWV- VXXXX)	Cypress Semicon- ductor	
63	1	N/A	N/A	LBL, PCBA Anti-Static Warning, 10mm X 10mm	Cypress Semicon- ductor	
64	1	N/A	N/A	Assembly Adhesive Label, Manufactur- ing ID	Cypress Semicon- ductor	
65	1	N/A	N/A	Kit QR code	Cypress Semicon- ductor	

A.7 Regulatory Compliance Information

The CY8CKIT-042 PSoC 4 Pioneer Kit has been tested and verified to comply with the following electromagnetic compatibility (EMC) regulations:

- EN 55022:2010 Class A Emissions
- EN 55024:2010 Class A Immunity



A.8 Migrating projects across different Pioneer series kits

All Cypress Pioneer series kits are Arduino Uno compatible and have some common on-board peripherals such as RGB LED, CapSense and User Switch. However, the pin mapping in each of the boards is different due to differences in pin functions of the PSoC device used. This guide lists the pin maps of the Pioneer series kits to allow for easy migration of projects across different kits.

In some cases, the pins available on the Pioneer kit headers are a superset of the standard Arduino Uno pins. For example J2 contains only 1 row of pins on the Arduino Uno pinout while it contains 2 rows of pins on many of the Pioneer series kits.

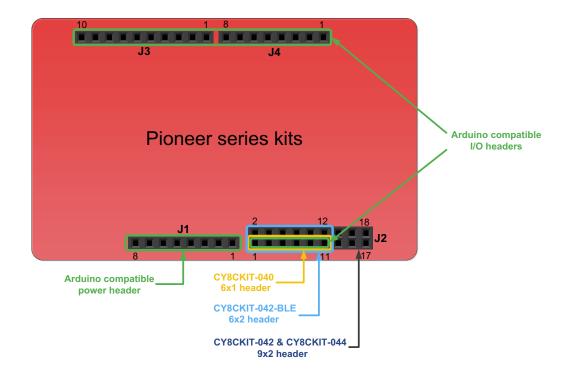


Figure A-1. Pioneer series kits pin map



A.8.1 Arduino Uno Compatible Headers

	J1 Arduino Compatible Header Pin Map								
Pin #	Arduino Pin		Pioneer series kits						
FIII #	Ardumo Pin	CY8CKIT-042	CY8CKIT-040	CY8CKIT-042-BLE	CY8CKIT-044				
1	VIN	VIN	VIN	VIN	VIN				
2	GND	GND	GND	GND	GND				
3	GND	GND	GND	GND	GND				
4	5V	V5.0	V5.0	V5.0	V5.0				
5	3.3V	V3.3	V3.3	V3.3	V3.3				
6	RESET	RESET	RESET	RESET	RESET				
7	IOREF	P4.VDD	P4.VDD	BLE.VDD	P4.VDD				
8	NC	NC	NC	NC	NC				

	J2 Arduino Compatible Header Pin Map									
Pin #	Arduino Pin		Pioneer	series kits						
FIII #	Alduno Pin	CY8CKIT-042	CY8CKIT-040	CY8CKIT-042-BLE	CY8CKIT-044					
1	A0	P2[0]	P0[0]	P3[0]	P2[0]					
2	-	P0[2] [*]	-	P2[0]	P2[6] [*]					
3	A1	P2[1]	P0[1]	P3[1]	P2[1]					
4	-	P0[3] [*]	_	P2[1] [*]	P6[5] [*]					
5	A2	P2[2]	P0[2] [*]	P3[2]	P2[2]					
6	-	P4_VDD	-	P2[2] [*]	P0[6] [*]					
7	A3	P2[3]	P0[4] [*]	P3[3]	P2[3]					
8	-	P1[5] [*]	-	P2[3] [*]	P4[4] [*]					
9	A4	P2[4]	P1[3]	P3[4]	P2[4]					
10	-	P1[4] [*]	_	P2[4] [*]	P4[5] [*]					
11	A5	P2[5]	P1[2]	P3[5]	P2[5]					
12	-	P1[3] [*]	_	P2[5] [*]	P4[6] [*]					
13	_	P0[0]	_	-	P0[0]					
14	_	GND	_	-	GND					
15	_	P0[1]	-	-	P0[1]					
16	-	P1[2] [*]	_	-	P3[4] [*]					
17	-	P1[0]	_	-	P0[7] [*]					
18	_	P1[1] [*]	_	-	P3[5] [*]					

* These pins are also used for on-board peripherals. See the tables in the On-Board Peripherals section below for details.



	J3 Arduino Compatible Header Pin Map								
Pin #	Arduino Pin	Pioneer series kits							
FIII #	Alduno Pin	CY8CKIT-042	CY8CKIT-040	CY8CKIT-042-BLE	CY8CKIT-044				
1	D8	P2[6]	P1[4]	P0[5]	P0[2]				
2	D9	P3[6]	P1[5]	P0[4]	P0[3]				
3	D10	P3[4]	P1[6]	P0[2]	P2[7]				
4	D11	P3[0]	P1[1] [*]	P0[0]	P6[0]				
5	D12	P3[1]	P3[1]	P0[1]	P6[1]				
6	D13	P0[6]	P1[7]	P0[3]	P6[2]				
7	GND	GND	GND	GND	GND				
8	AREF	P1[7]	NC	VREF	P1[7]				
9	SDA	P4[1]	P1[3]	P3[4]	P4[1]				
10	SCL	P4[0]	P1[2]	P3[5]	P4[0]				

* These pins are also used for on-board peripherals. See the tables in the On-Board Peripherals section below for connection details.

	J4 Arduino Compatible Header Pin Map								
Pin #	Arduino Pin	Pioneer series kits							
FIII#		CY8CKIT-042	CY8CKIT-040	CY8CKIT-042-BLE	CY8CKIT-044				
1	D0	P0[4]	P0[5]	P1[4]	P3[0]				
2	D1	P0[5]	P0[6]	P1[5]	P3[1]				
3	D2	P0[7] [*]	P0[7]	P1[6]	P1[0]				
4	D3	P3[7]	P3[2] [*]	P1[7]	P1[1]				
5	D4	P0[0]	P0[3]	P1[3]	P1[2]				
6	D5	P3[5]	P3[0]	P1[2]	P1[3]				
7	D6	P1[0]	P1[0]	P1[1]	P5[3]				
8	D7	P2[7]	P2[0] [*]	P1[0]	P5[5]				

* These pins are also used for on-board peripherals. See the tables in the On-Board Peripherals section below for connection details.



A.8.2 On-Board Peripherals

	CapSense Pin Map								
			Pioneer series kits						
Pin #	Description	CY8CKIT-042	CY8CKIT-040	CY8CKIT-042-BLE	CY8CKIT-044				
		(Linear Slider)	C10CK11-040	(Linear Slider)	(Gesture Pad)				
1	CSS1	P1[1]	-	P2[1]	P4[4]				
2	CSS2	P1[2]	-	P2[2]	P4[5]				
3	CSS3	P1[3]	-	P2[3]	P4[6]				
4	CSS4	P1[4]	-	P2[4]	P3[4]				
5	CSS5	P1[5]	-	P2[5]	P3[5]				
6	CMOD	P4[2]	P0[4]	P4[0]	P4[2]				
7	CTANK	P4[3]	P0[2]	P4[1]	P4[3]				

	Proximity header Pin Map							
Pin #	Description		Pioneer series kits					
FIII #	Description	CY8CKIT-042	CY8CKIT-040	CY8CKIT-042-BLE	CY8CKIT-044			
1	PROXIMITY	-	P2[0]	P2[0]	P3[7]			
2		_	_	-	P3[6]			

	RGB LED Pin Map								
Pin #	Color		Pioneer series kits						
FIII #	Color	CY8CKIT-042	CY8CKIT-040	CY8CKIT-042-BLE	CY8CKIT-044				
1	Red	P1[6]	P3[2]	P2[6]	P0[6]				
2	Green	P0[2]	P1[1]	P3[6]	P2[6]				
3	Blue	P0[3]	P0[2]	P3[7]	P6[5]				

	User Switch Pin Map							
Pin #	Description	Pioneer series kits						
FIII #		CY8CKIT-042	CY8CKIT-040	CY8CKIT-042-BLE	CY8CKIT-044			
1	SW2	P0[7]	-	P2[7]	P0[7]			

Revision History



Document Revision History

CY8CKIT-042 PSoC® 4 Pioneer Kit Guide Revision History

Document	Document Title: CY8CKIT-042 PSoC® 4 Pioneer Kit Guide						
Document	Document Number: 001-86371						
Revision	ECN#	Issue Date	Origin of Change	Description of Change			
**	3978908	04/23/2013	ANCY	Initial version of kit guide.			
*A	3981609	04/25/2013	ANCY	Minor changes across the guide.			
*B	4008979	05/23/2013	RKAD	Minor changes across the guide. Updated Introduction chapter on page 7: Updated "Kit Contents" on page 7:			
				Updated Figure 1-1. Updated Advanced Topics chapter on page 66: Added "PSoC 5LP Factory Program Restore Instructions" on page 97.			
*C	4107338	08/23/2013	SASH	Minor changes across the guide. Updated Code Examples chapter on page 47: Updated Figure 5-2. Updated Figure 5-3.			
*D	4202835	11/26/2013	SASH	Updated PSoC Creator images. Added figure captions. Updated Introduction chapter on page 7: Updated "Additional Learning Resources" on page 10: Updated PSoC Creator training web link. Updated Code Examples chapter on page 47: Modified the CapSense code example.			
*E	4757883	05/07/2015	SASH / MSUR	Updated Introduction chapter on page 7: Updated "Additional Learning Resources" on page 10: Updated description. Updated "PSoC Creator Code Examples" on page 12: Updated Figure 1-4.			



Document Number: 001-86371					
Revision	ECN#	Issue Date	Origin of Change	Description of Change	
				Updated Software Installation chapter on page 15:	
				Updated "Install Kit Software" on page 15:	
				Updated description.	
				Updated "Install Software" on page 16:	
				Updated description.	
				Updated "Develop Code Fast and Easy with Code Examples" on page 17:	
				Updated Figure 2-3.	
				Updated Figure 2-4.	
				Updated "Open an Example Project in PSoC Creator" on page 19:	
				Updated Figure 2-6.	
				Updated Kit Operation chapter on page 21:	
		0.5/0.7/0.0/5	SASH /	Updated "Pioneer Kit USB Connection" on page 22:	
*E (cont.)	4757883	05/07/2015	MSUR	Updated Table 3-1:	
				Updated entire table.	
				Removed figure "KitProg Driver Installation".	
				Updated Figure 3-2.	
				Updated "Programming and Debugging PSoC 4" on page 23:	
				Updated "Using CY8CKIT-002 MiniProg3 Programmer and Debugger" o page 25:	
				Updated Figure 3-8.	
				Updated description.	
				Updated Hardware chapter on page 31:	
				Updated "Functional Description" on page 34:	
				Updated "Arduino Compatible Headers (J1, J2, J3, J4, and J12 - unpop lated)" on page 40:	
				Updated description.	



Document Number: 001-86371						
Revision	ECN#	Issue Date	Origin of Change	Description of Change		
	4757883			Updated Code Examples chapter on page 47:		
				Updated description.		
				Updated "Project: Blinking LED" on page 50:		
				Updated "Hardware Connections" on page 50:		
				Updated Table 5-1:		
				Updated details in "Pin Name" column.		
				Updated Figure 5-8.		
				Updated "Flow Chart" on page 51:		
				Updated description.		
				Updated "Project: PWM" on page 52:		
				Updated "Hardware Connections" on page 52:		
				Updated Table 5-2:		
				Updated details in "Pin Name" column.		
			SASH /	Updated Figure 5-12.		
				Updated "Project: Deep Sleep" on page 54:		
				Updated "Hardware Connections" on page 54:		
				Updated Table 5-3:		
'E (cont.)		05/07/2015		Updated details in "Pin Name" column.		
			MSUR	Updated Figure 5-15.		
				Updated "Project: CapSense" on page 56:		
				Updated "CapSense (Without Tuning)" on page 56:		
				Updated "Hardware Connections" on page 57:		
				Updated Table 5-4:		
				Updated details in "Pin Name" column.		
				Updated Figure 5-18.		
				Updated "CapSense (With Tuning)" on page 59:		
				Updated "Launching Tuner GUI" on page 60:		
				Updated Figure 5-22.		
				Updated Figure 5-23.		
				Updated "Verify Output" on page 62:		
				Updated description.		
				Updated Figure 5-25.		
				Updated Figure 5-26.		
				Updated Figure 5-27.		
				Updated Figure 5-28.		



Document Number: 001-86371						
Revision	ECN#	Issue Date	Origin of Change	Description of Change		
	4757883		SASH / MSUR	Updated Advanced Topics chapter on page 66:		
				Updated "Using PSoC 5LP as USB-UART Bridge" on page 66:		
				Updated description.		
				Updated Figure 6-1.		
				Updated Figure 6-2.		
				Updated Figure 6-4.		
				Updated Figure 6-5.		
				Updated Figure 6-6.		
				Updated Figure 6-7.		
				Updated "Using PSoC 5LP as USB-I2C Bridge" on page 79:		
				Updated description.		
				Updated Figure 6-19.		
				Updated Figure 6-20.		
				Updated Figure 6-22.		
		05/07/2015		Updated Figure 6-23.		
				Added Figure 6-24.		
				Updated Figure 6-25.		
E (cont.)				Updated "Developing Applications for PSoC 5LP" on page 88:		
				Updated "Building a Bootloadable Project for PSoC 5LP" on page 88:		
				Updated description.		
				Updated Figure 6-33.		
				Updated Figure 6-35.		
				Added Figure 6-39.		
				Updated Figure 6-40.		
				Updated "Building a Normal Project for PSoC 5LP" on page 96:		
				Updated Figure 6-45.		
				Updated "PSoC 5LP Factory Program Restore Instructions" on page 93		
				Updated "PSoC 5LP is Programmed with a Bootloadable Application" of page 97:		
				Updated "Restore PSoC 5LP Factory Program Using PSoC Program- mer" on page 97:		
				Updated description.		
				Updated "PSoC 5LP is Programmed with a Standard Application" on page 102:		
				Updated description.		
				Added "Using µC/Probe Tool" on page 104.		



Document Title: CY8CKIT-042 PSoC® 4 Pioneer Kit Guide							
Document Number: 001-86371							
Revision	ECN#	Issue Date	Origin of Change	Description of Change			
*E (cont.)	4757883	05/07/2015	SASH / MSUR	Updated Appendix chapter on page 113:			
				Updated "CY8CKIT-042 Schematics" on page 113:			
				Updated entire section.			
				Updated "Use of Zero-ohm Resistors and No Load" on page 120:			
				Updated table.			
				Added "Migrating projects across different Pioneer series kits" on page 124.			
*F	4897811	09/14/2015	SRDS	Updated images and fixed hyperlinks.			
*G	5201185	04/01/2016	RKAD	Sunset review; no content updates			