

KitProg2 User Guide

Doc. # 002-10738 Rev. *I

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



Copyrights

© Cypress Semiconductor Corporation, 2016–2018. 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, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. 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, PSoC Creator, 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.

Contents



1.	Introdu	ction	5
	1.1	Switching between KitProg2 Modes	7
	1.2	Acronyms	
2.	Ecosys	tem	10
3.	KitProg	2 Mode Programmer and Debugger	12
	3.1	KitProg2 Driver Installation	12
	3.2	Programming Using PSoC Creator	
	3.3	Debugging Using PSoC Creator	16
	3.4	Programming Using PSoC Programmer	17
	3.5	Updating the KitProg2 Firmware	17
4.	CMSIS-	DAP Mode Programming and Debugging	20
	4.1	Programming and Debugging using μV ision	20
5.	Mass S	torage Programmer	27
	5.1	Enter or Exit the Mass Storage Programmer Mode	27
	5.2	Programming Using the Mass Storage Programmer	
	5.3	Frequently Asked Questions on KitProg2 Mass Storage Programmer	
6.	USB-U/	ART Bridge	30
7.	USB-I2	C Bridge	41
8.	USB-SF	PI Bridge	52
9.	Develo	ping Applications for PSoC 5LP	58
	9.1	Building a Custom Bootloadable Application for KitProg2	58
	9.2	Building a Normal Project for PSoC 5LP	
10). Troub	leshooting the KitProg2	71
	10.1	KitProg2 Status LED Indication	71
	10.2	-	
		10.2.1 PSoC 5LP is Programmed with a Custom Application	
		10.2.1.1 Restore PSoC 5LP Factory Program	
		Using PSoC Programmer	
		10.2.2 Restore PSoC 5LP KitProg2 using MiniProg3	76



Α.	Appendi	x		78
	A.1	Pin As	signments	78
		A.1.1	PSoC 5LP GPIO and Custom Application Header (J8 and J11)	
			for CY8CKIT-041-40XX and CY8CKIT-041-41XX	78
		A.1.2	PSoC 5LP GPIO and Custom Application Header (J16 and J11)	
			for CY8CKIT-048	79
		A.1.3	PSoC 5LP GPIO and Custom Application Header (J6 and J7)	
			for CY8CKIT-145-40XX	79
Re	vision His	story		80

Revision History

1. Introduction



The KitProg2 is an onboard programmer/debugger with USB-UART, USB-I2C and USB-SPI Bridge functionality. It is an update to the existing KitProg used for programming and debugging the target device. The KitProg2 is integrated onto most PSoC[®] development kits. This user guide provides comprehensive information on how to use the KitProg2 with PSoC development kits. Figure 1-1 shows the KitProg2 ecosystem. The Cypress PSoC 5LP device is used to implement the KitProg2 functionality.

Figure 1-1. KitProg2 Ecosystem





flash

KitProg2 is an enhancement over KitProg. It follows the dual-image bootloading approach; the KitProg2 firmware is the first image and a custom application can be loaded as a second image. To learn more about the concept of dual-image bootloading, refer to application note AN73854 -PSoC 3, PSoC 4, and PSoC 5LP Introduction to Bootloaders.

Figure 1-2. KitProg2 Dual-Image Bootloader Architecture



Figure 1-2 shows the KitProg2 flash architecture based on the concept of dual-image bootloading. If you are building custom applications for PSoC 5LP in KitProg2, only the 'Custom Application Image' flash area can be bootloaded with the corresponding cyacd file. The KitProg2 image cannot be bootloaded. It can only be restored to factory settings as described in Restore PSoC 5LP Factory Program Using PSoC Programmer on page 74.

The KitProg2 image contains SWD, CMSIS-DAP, and Mass Storage Programmer functionality to program a target PSoC, and USB-UART/USB-I2C/USB-SPI Bridge functionality. The custom application image can be any application that PSoC 5LP can execute. To create custom applications, refer to the Developing Applications for PSoC 5LP chapter on page 58.



1.1 Switching between KitProg2 Modes

There are three types of kits that use KitProg2 programmers. These are: (1) prototyping kits (also known as stamp boards) which have a single mode switch and a single amber status LED; (2) development kits (also known as pioneer kits) which have a single mode switch and 3 status LEDs; and (3) development kits which have two mode switches and three status LEDs.

Figure 1-3 shows how to switch between modes in KitProg2. Part (a) of Figure 1-3 is valid for all kits with a single mode switch. This figure illustrates the switching workflow using the CY8CKIT-041-40XX kit as an example, which has SW3 as the mode switch. Part (b) of Figure 1-3 is valid for all kits with dual mode switches. This figure illustrates the switching workflow using the CY8CKIT-062 BLE Pioneer Kit. In either case the input on the mode switch or switches is evaluated; depending on the current mode of operation, the next mode of operation is ascertained.

On power-on reset or reset, PSoC 5LP enters bootloader entry mode. If the mode switch SW3 was pressed while the USB connector was plugged in and then released, KitProg2 enters bootloader mode. If the mode switch SW3 was not pressed, then depending on the current mode of operation, PSoC 5LP will enter PPCOM mode, Mass Storage Programming/CMSIS-DAP mode, or the custom application.

As illustrated in Figure 1-3 part (a) switching between KitProg2 and Mass Storage Programming/ CMSIS-DAP mode can be achieved by pressing and releasing the mode switch within five seconds. Similarly, switching to the custom application from the PPCOM or Mass Storage Programming/ CMSIS-DAP mode can be achieved by pressing and holding the mode switch for more than five seconds and then releasing. Switching from the custom mode back to PPCOM or Mass Storage Programming/CMSIS-DAP modes is dependent on the custom application implementation.

In order to recognize the various state changes the Amber LED shows different effects. When in KitProg2 Mode the Amber LED lights up, when in Mass Storage/CMSIS-DAP mode the Amber LED turns off and when in bootloader mode the Amber LED shows a blinking effect.







b) For Two Mode Switch Development Kits



Note: In order to switch back from custom mode application to CMSIS-DAP/Mass Storage or KitProg2 within the firmware, refer Developing Applications for PSoC 5LP on page 58.

Note: After pressing and releasing the mode switch, it may take a few seconds for the kit to re-enumerate in the new mode. While enumeration is taking place, none of the status LEDs will be ON.



1.2 Acronyms

Acronym	Definition
BCP	Bridge Control Panel
BLE	Bluetooth Low Energy
CMSIS-DAP	Cortex Microcontroller Software Interface Standard Debug Access Protocol
GPIO	General-Purpose Input/Output
HID	Human Interface Device
I2C	Inter-Integrated Circuit
IDE	Integrated Development Environment
JTAG	Joint Test Action Group
LED	Light-Emitting Diode
MISO	Master-In-Serial-Out
MOSI	Master-Out-Serial-In
NVL	Non Volatile Latch
PC	Personal Computer
PPCOM	PSoC Programmer Component Object Module
PSoC	Programmable System-on-Chip
RAM	Random Access Memory
ROM	Read-Only Memory
SCB	Serial Communication Block
SCL	Serial Clock Line
SDA	Serial Data Line
SPI	Serial Peripheral Interface
SWD	Serial Wire Debug
UART	Universal Asynchronous Receiver Transmitter
UDB	Universal Digital Block
USB	Universal Serial Bus
XRES	External Reset





Table 2-1 lists the development kits that use the KitProg2. Table 2-2 lists the prerequisite Cypress software needed to use the KitProg2.

Table 2-1. Development Kits Supported by KitProg2

Development Kits	Target Device	KitProg2 onboard mode switches
CY8CKIT-041-40XX PSoC 4 S-Series Pioneer Kit	PSoC 4000S	Single Switch
CY8CKIT-041-41XX PSoC 4100S Pioneer Kit	PSoC 4100S	Single Switch
CY8CKIT-048 PSoC Analog Coprocessor Pioneer Kit	PSoC Analog Coprocessor	Single Switch
CY8CKIT-145-40XX PSoC 4 S-Series Prototyping Kit	PSoC 4000S	Single Switch
CY8CKIT-146 PSoC 4200DS Prototyping Kit	PSoC 4200DS	Single Switch
CY8CKIT-147 PSoC 4100PS Prototyping Kit	PSoC 4100PS	Single Switch
CY8CKIT-149 PSoC 4100S Plus Prototyping Kit	PSoC 4100S Plus	Single Switch
CY8CKIT-062 BLE Pioneer Kit	PSoC 6	Dual Switch
CY8CKIT-062-WiFi-BT PSoC 6 WiFi-BT Pioneer Kit	PSoC 6 WiFi-BT	Dual Switch

Table 2-2. Prerequisite Software for KitProg2 Operation

Functionality	Pre-requisite Software	Download Link/Remarks
Programmer	PSoC Programmer	www.cypress.com/psocprogrammer
Debugger	PSoC Creator	www.cypress.com/psoccreator
USB-I2C Bridge	Bridge Control Panel (BCP)	Installed along with PSoC Programmer
USB-SPI Bridge	Bridge Control Panel (BCP)	Installed along with PSoC Programmer
USB-UART Bridge	Terminal Emulator Program	Any terminal emulator program can be used such as HyperTerminal (available as part of Microsoft Windows XP installation) or PuTTY (available on www.putty.org)



KitProg2 supports different speeds for communication interfaces. Table 2-3 summarizes the KitProg2 operating modes.

Table 2-3.	KitProg2	Operating	Modes
------------	----------	-----------	-------

Functionality	Supported Speed	Units
Programmer	1.6	MHz
USB-UART Bridge	1200, 2400, 4800, 9600, 19200, 38400, 57600, and 115200	Baud
USB-I2C Bridge	50, 100, 400, 1000	kHz
USB-SPI Brdige	50–6000	kHz

This document assumes that you know the basics of using PSoC Creator[™]. If you are new to PSoC Creator, refer to the documentation in the PSoC Creator home page. You can also refer to the following application notes to get started with PSoC devices:

- Getting Started with PSoC[®] 4
- Getting Started with PSoC[®] 4 BLE
- Getting Started with PSoC[®] 5LP
- Getting Started with CapSense[®]

3. KitProg2 Mode Programmer and Debugger



This section explains how the KitProg2 programmer/debugger is integrated onto the PSoC development kits. The KitProg2 supports the development kits listed in Table 2-1 on page 10. This section uses the PSoC 4 S-Series Pioneer Kit as an example.

KitProg2 uses two types of programming/debugging interfaces – PPCOM and CMSIS-DAP. The PPCOM interface supports Cypress tool chains such as PSoC Creator and PSoC Programmer. The PPCOM interface provides additional options such as programming NVLs, which are not available via a standard CMSIS-DAP interface.

CMSIS-DAP is an alternative programming/debugging interface in which the KitProg2 can be used with third-party tool chains to program/debug the target. This mode is selected when you press and release the mode switch for less than two seconds.

The amber status LED stays ON if the KitProg2 is in PPCOM interface mode and it turns off at a rate of 1 Hz if the KitProg2 is in CMSIS-DAP/Mass Storage Programming mode.

Refer to Switching between KitProg2 Modes on page 7 to understand the behavior of the status LED.

Note: PPCOM is the abbreviation for PSoC Programmer Component Object Module. It is the programming/debugging interface provided in Cypress' PSoC Creator and PSoC Programmer.

3.1 KitProg2 Driver Installation

The kit is powered from a computer over the USB interface. It enumerates as a composite device. The USB drivers required for enumeration are part of the kit installer and should be appropriately installed for correct operation.

Figure 3-1 shows the driver installation in PPCOM programming mode and Figure 3-2 shows the driver installation in CMSIS-DAP programming mode.

Figure 3-1. KitProg2 Driver Installation in PPCOM Programming Mode

Driver Software Installation		×
Your device is ready to use		
USB Composite Device USB Input Device KitProg2 USB-UART (COM50)	Ready to use Ready to use Ready to use	
		Close



0 0		0
Driver Software Installation		×
Your device is ready to use		
USB Composite Device USB Input Device USB Mass Storage Device CYPRESS KitProg USB Device	Ready to use Ready to use Ready to use Ready to use	
		Close

Figure 3-2. KitProg2 Driver Installation in CMSIS-DAP Programming Mode

3.2 **Programming Using PSoC Creator**

- 1. Connect a USB cable to the USB connector, J6, as shown in Figure 3-3. If you are connecting the kit to your PC for the first time, it enumerates as a USB composite device and installs the required driver software.
- 2. Verify the KitProg2 is in PPCOM mode (Amber LED is ON). See Switching between KitProg2 Modes on page 7 for details.

Figure 3-3. Connect USB Cable to J6 (Pioneer Kits)



- 3. Launch PSoC Creator from Start > All Programs > Cypress > PSoC Creator <version> > PSoC Creator <version>.
- Select File > Open > Project/Workspace in PSoC Creator and browse to the desired project. You may also select File > Code Example to browse through a library of existing example projects. See PSoC Creator User Guide for details.



5. Select Build > Build Project or press [Shift] [F6] to build the project, as shown in Figure 3-4.

Figure 3-4. Build an Example Project

	Build CY8CKIT_041_Proximity_Sensing	g Shift+F6
	Clean CY8CKIT_041_Proximity_Sensin	9
) 🖽	Clean and Build CY8CKIT_041_Proxim	ity_Sensing
à	<u>C</u> ancel Build	Ctrl+Break
P 🕸	Compile <u>F</u> ile	Ctrl+F6
1	Generate Application	
in 🛅	Generate Project Datasheet	

6. If there are no errors during build, program the PSoC 4000S device on the kit by choosing Debug
 > Program or pressing [Ctrl] [F5], as shown in Figure 3-5.

Figure 3-5. Programming Device from PSoC Creator





7. If the device is already acquired, programming will complete automatically – the result will appear in the PSoC Creator status bar at the bottom left of the screen. If the device is yet to be acquired, the Select Debug Target window will appear. Select KitProg2/<serial number> and click the Port Acquire button, as shown in Figure 3-6.

Figure 3-6. Port Acquire

Select Debug Target	? ×
■-5 KitProg2/1D1A18EB02105400	KitProg2/1D1A18EB02105400
└-	POWER = 3 VOLTAGE_ADC = 4812 FREQUENCY = 2000000 PROTOCOL = SWD
	KitProg2 Version 0.07 [HW Rev.0x01]
Show all targets	Port Setting Port Acquire
	ОК

 After the device is acquired, it is shown in a tree structure below the KitProg2/<serial number>. Click the Connect button and then OK to exit the window and start programming, as shown in Figure 3-7.



Figure 3-7. Connect Device from PSoC Creator and Program



3.3 Debugging Using PSoC Creator

To debug the project using PSoC Creator, follow steps 1 to 5 from section Programming Using PSoC Creator on page 13. Then, follow these steps:

1. Click the **Debug** icon or press **[F5]**, as shown in Figure 3-8. Alternatively, you can select **Debug** > **Debug**. This programs the device and starts the debugger.

Figure 3-8. Debug Option in PSoC Creator

Deb	bug	Tools	Window	<u>H</u> elp	
	Win	ndows			•
	Pro	gram			Ctrl+F5
ž	Sel	ect Debu	g <u>T</u> arget		
账	Del	bug			F5
贫	Del	bug wit <u>h</u>	out Prograr	nming	Alt+F5
	<u>A</u> tt	ach to Ri	unning Targ	jet	
5	Tog	ggle Brea	kpoint		F9
	Ne	w <u>B</u> reakp	oint		•
0	Del	ete All B	rea <u>k</u> points	Ctrl	+Shift+F9
0	Ena	able All B	reakpoints		

2. When PSoC Creator enters the Debug mode, use the buttons on the toolbar or keyboard shortcuts to debug your project.

For more details on using the debug features, refer to the PSoC Creator Help. Select **Help** > **PSoC Creator Help Topics** in the PSoC Creator menu. In the PSoC Creator Help window, locate **Using the Debugger** section in the **Contents** tab, as shown in Figure 3-9.



Figure 3-9. Using the PSoC Creator Debugger

The debuoner understands features that are built into renorammino languages and their associated libraries. With the debuoner you can break (suspend) execution of your t



3.4 Programming Using PSoC Programmer

PSoC Programmer (3.24.2 or later) can be used to program existing *.hex* files into the kit. To do this, follow these steps.

- 1. Connect the kit to your PC and open PSoC Programmer from Start > All Programs > Cypress > PSoC Programmer <version> > PSoC Programmer <version>.
- 2. Click the File Load button at the top left corner of the window. Browse to the desired .hex file and click Open. For PSoC 4000S devices, the .hex file is located at: <Project Directory>\<Project Name.cydsn>\CortexM0p\<Compiler Name and Version>\<Debug> or <Release>\<Project Name.hex>.
- 3. Click the **KitProg2/<serial number>** in the **Port Selection** list to connect the kit to your computer.
- 4. Click the Program button to start programming the kit with the selected file.

Note: If the *.hex* file does not match the selected device, PSoC Programmer will display a device mismatch error and terminate programming. Ensure that you have selected the correct *.hex* file.

5. When programming is completed successfully, indicated by a PASS message on the status bar, the kit is ready for use. Close PSoC Programmer.

3.5 Updating the KitProg2 Firmware

The KitProg2 firmware generally does not require any update. If an update is required, then PSoC Programmer will display a warning message when the kit is connected to it, as shown in Figure 3-10.

WARNING!

Figure 3-10. KitProg2 Firmware Update Warning



Click **OK** to close the window. On closing the warning window, the Actions and Results window displays: "Please navigate to the Utilities tab and click the Upgrade Firmware button", as shown in Figure 3-11.

To update the KitProg2, go to the **Utilities** tab on PSoC Programmer and click **Upgrade Firmware**, as shown in Figure 3-11.

PSoC Programmer
File View Options Help
📂 · 🗼 💿 BB 🛃 🖹 🗋 🕲
Port Selection IVilities JTAG
KitProg2/0B0C05B3031054 Upgrade Firmware Click to upgrade connected device's firmware
Erase Block Click to erase user specific flash block
Device Family
CY8C4Doc-S v
Device
CY8C4045AZI-S413 -
Actions Results
Please navigate to the Utilities tab and click the Upgrade Firmware button
Port Opened with Warnings at 3:39:31 PM KitProg2 version Expecting 1.01, but found 1.00.
Opening Port at 3:39:28
PM Connected at 3:39:28 PM KitProg2/0B0C05B303105400
Some Bold ab Stores In Altilog2/ BBCCOBBCCIDING
For Help, press F1 Connected Connected

Figure 3-11. Upgrade Firmware in PSoC Programmer



On successful upgrade, the Actions and Results window displays the firmware update message with the KitProg2 version, as shown in Figure 3-12.

Figure 3-12. Firmware Updated in PSoC Programmer

PSoC Programmer			
File View Options Hel	p		
🖆 · 🗼 🥥 🕒			
Port Selection	Programmer Utilitie	es JTAG	
KitProg2/0A2116F7000454(Upgrade Firmwan Erase Block		de connected device's firmware user specific flash block
Device Family			
CY8C40xx-S 👻			
Device			
CY8C4045AZI-S413 -			KitProg2 firmware version
Actions	Results		
	-	Version 1.01	[HW Rev.0x01]
Firmware Update Fini at 1:30:02 PM	shed Succeeded		KitProg2 firmware upgrade message
	Verifying		
	Upgrading Initializ		
Firmware Upgrade Sta at 1:29:45 PM Firmware Upgrade Requested at 1:29:45	rted	Ing	
For Help, press F1			PASS Powered Connected

4. CMSIS-DAP Mode Programming and Debugging



This section explains how the CMSIS-DAP programmer/debugger is integrated onto the PSoC development kits. The KitProg2 supports the development kits listed in Table 2-1. This section uses the PSoC 4 S-Series Pioneer Kit as an example.

The KitProg2 programmer and debugger (PPCOM), KitProg2 USB-I2C Bridge, and KitProg2 USB-UART Bridge functionalities are not available in this configuration.

CMSIS-DAP is an alternative programming/debugging interface in which the KitProg2 can be used with third-party tool chains to program/debug the target. This mode is selected when you press and release the mode switch for less than two seconds.

The amber status LED stays ON if the KitProg2 is in PPCOM interface mode and it turns off at a rate of 1 Hz if the KitProg2 is in CMSIS-DAP/Mass Storage Programming mode.

Refer to Switching between KitProg2 Modes on page 7 to understand the behavior of status LED.

4.1 **Programming and Debugging using µVision**

CMSIS-DAP mode can be used for programming from many third party IDEs. As an example, the steps to program using μ Vision are shown below.

To use KitProg2 in CMSIS-DAP mode to program using µVision, do the following:

1. Connect the kit to PC and enter into CMSIS-DAP mode. To enter into CMSIS-DAP mode, refer Switching between KitProg2 Modes on page 7.

Note: The KitProg2 should be in the CMSIS-DAP programming mode (press mode switch SW3 on CY8CKIT-041-40XX for less than two seconds to change modes). In this mode, the amber LED switches OFF.

Note: in KitProg2 1.04 version, this LED will show breathing effect in Mass Storage or CMSIS-DAP mode.



2. Open the project in PSoC Creator and build the project. Right-click the project in the Workspace Explorer and select **Export to IDE**, as shown in Figure 4-1.

Figure 4-1. Export to IDE

🖓 🔁		
	ace 'CapSense_LowPower' (1 Pro	
E. Droi	Set As Active Project	BC4045471-S.
-1	Add	•
	Build CapSense_LowPower	
₽1 —	Clean CapSense_LowPower	
	Clean and Build CapSense_Low	Power
	Updat <u>e</u> Components	
Eb	<u>С</u> ору	Ctrl+C
12	Paste	Ctrl+V
	Save CapSense_LowPower As	
	Remove From Workspace	
	Rename	F2
	Unload Project	
	Dependencies	
	Build Order	
	Device Selector	
	Archive Workspace/Project	
۲	Export to IDE (CapSense_LowPo	ower)
9	Project Resources	

- 3. Select the IDE as shown in Figure 4-2.
- Figure 4-2. Select IDE





4. Select the tool chain as shown in Figure 4-3.

Figure 4-3. Select Tool Chain

IDE Export Wizard - CapSense_LowPower	? <mark>x</mark>
Select Toolchain Choose the desired tool chain to build with in $\mu Vision$	
Select target toolchain: ARM MDK Generic ARM MDK Generic ARM GCC Generic	
< Back Next > Cancel	Help

5. Select the project files as shown in Figure 4-4. The next window that appears is **Review export** details. Click **Export**.

Figure 4-4. Select Project Files

IDE Export Wizard - Ca	pSense_LowPower	? <mark>X</mark>					
Application Files Select the files you wish to export. They will be added to the µVision project.							
Select All Uns	elect All						
Name	Path						
 ✓ main.c ✓ cyapicallbacks.h 	G:\Projects\Kit_Prog\Projects\Design01\CapSense_LowPow G:\Projects\Kit_Prog\Projects\Design01\CapSense_LowPow						
•	m	+					
< [Back Next > Cancel	Help					

6. Open the μ Vision project in the μ Vision IDE tool. Keil μ Vision version 4.74.0.22 is used in this illustration. Build the project in μ Vision.



- 7. Open the PSoC Programmer installation folder and look for the path Programmer\3rd_Party _Configuration_Files\CY8C40xx\Prog_Algorithm. Note: The default path location of PSoC programmer installation folder is C:\Program Files (x86)\Cypress\Programmer.
- 8. Copy the CY8C40xx.FLM file to the $\ARM\Flash$ folder inside the installation directory of Keil μ Vision. This will typically look similar to: C:\Keil\ARM\Flash\
- 9. Select Project > Options for Target "project_name".

Figure 4-5. Option for Target in µVision

rog\Proje	cts\Desi	gn01\Cap	Sense_LowPo	wer.cyd	Isn\Cap	Sense_Low	Power.uvproj - µVision4	
Project	Flash	Debug	Peripherals	Tools	SVCS	Window	Help	
Nev Op Sav	w Multi-l en Proje	ct : in μVisio	orkspace n4 format					
Exp Ma	ort nage							•
	e ct Devic nove Iter		et 'CapSense_	LowPow	/er'			
💦 Op	tions for	Target 'C	apSense_LowF	ower'				Alt+F7
Cla	an taraa	÷						

10. Make sure to set appropriate values for ROM and RAM areas of the target device as shown in Figure 4-6.

Figure 4-6. Target Tab

Options for Target 'CapSense_LowPower'						
Device Target Output Listing User C/C++ Asm	Linker Debug Utilities					
Cypress CY8C4045AZI-S413 <u>X</u> tal (MHz): 16.0 Code Generation						
Operating system: None	Use Cross-Module Optimization					
System-Viewer File (.Sfr):	☑ Use MicroLIB					
Use Custom SVD File						
Read/Only Memory Areas	Read/Write Memory Areas					
default off-chip Start Size Startup	default off-chip Start Size NoInit					
□ ROM1: □ C	□ RAM1: □					
□ ROM2: □ C	RAM2:					
E ROM3: C	RAM3:					
on-chip	on-chip					
IROM1: 0x0 0x4000 €	✓ IRAM1: 0x20000000 0x800 □					
IROM2: C	IRAM2:					
ОК Сан	cel Defaults Help					



11. Select the Utilities tab and select CMSIS-DAP Debugger, as shown in Figure 4-7.

Figure 4-7. Utilities Tab in µVision.

Options for Target 'CapSense_LowPower'
Device Target Output Listing User C/C++ Asm Linker Debug Utilities
Configure Flash Menu Command
Use Target Driver for Flash Programming
CMSIS-DAP Debugger Settings Update Target before Debugging
Init File: ULINK2/ME Cortex Debugger
ULINK Pro Cortex Debugger
Command: SiLabs UDA Debugger
Arguments: CMSIS-DAP Debugger
Fast Models Debugger Cypress MiniProg3/KitProg v3.3
Configure Image File Processing (FCARM):
Output File: Add Output File to Group:
Generated_Header 💌
Image Files Root Folder:
OK Cancel Defaults Help

12. A programming algorithm must be added in the IDE to program PSoC 4. Click the Settings option. The Cortex-M Target Driver Setup window will open as shown in Figure 4-8. Go to the Flash Download tab and click the Add button as shown in Figure 4-9. Select the option corresponding to PSoC 4 in the programming algorithm.

Figure 4-8.	Cortex-M	Target	Driver	Setup
-------------	----------	--------	--------	-------

Options for Target 'CapSense_L	.owPower'			<u> </u>
Cortex-M Target Driver Setup	an (100-)	has [] seture [] is		×
Debug Flash Download		RAM for	Algorithm	1
C Erase Full Chip C Erase Sectors C Do not Erase	 Program Verify Reset and R 	Start:	Dx20000200 Size: Dx0600	
Programming Algorithm	, , , , , , , , , , , , , , , , , , , ,			
Description	Device Size	Device Type	Address Range	
		Start:	Size:	
	Add	Remove		
	ОК	Cance	el	Help
	ОК	Cancel	Defaults	Help



Init File: .\FLASH.ini	ger _	Settings	Update Target befo	re Debugging
x-M Target Driver Setup	_			
bug Flash Download				
Download Function		RAM for	Al 31	
0.5.5.00	Program	- RAMITOR	Avgonthm	
Erase Full Chip Frase Sectors	Verify	Start-	0x20000200 Size:	x0600
C Do not Erase	Reset and F		5120. p	
	It Reset and I			
Programming Algorithm				
Description	Device Size	Device Type	Address Range	
				X
Add Flash Programming Alg	orithm			
	Deck Con			
Description	Flash Size	Device Type	Origin	*
Description ATSAMR21 128kB Flash	128k	On-chip Flash	Origin MDK Core	^
				^
ATSAMR21 128kB Flash	128k	On-chip Flash	MDK Core	^
ATSAMR21 128kB Flash ATSAMR21 256kB Flash	128k 256k	On-chip Flash On-chip Flash	MDK Core MDK Core	^
ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 64kB Flash	128k 256k 64k	On-chip Flash On-chip Flash On-chip Flash	MDK Core MDK Core MDK Core	^
ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 64kB Flash CC2538xx 128 KB	128k 256k 64k 128k	On-chip Flash On-chip Flash On-chip Flash On-chip Flash	MDK Core MDK Core MDK Core MDK Core	^
ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 64kB Flash CC2538xc 128 KB CC2538xc 256 KB	128k 256k 64k 128k 256k	On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash	MDK Core MDK Core MDK Core MDK Core MDK Core	^
ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 64kB Flash CC2538x 128 KB CC2538x 256 KB CC2538x 512 KB	128k 256k 64k 128k 256k 512k 16k	On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash	MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core	
ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 54kB Flash CC2538x 256 KB CC2538x 512 KB CC2538x 512 KB CC2538x 512 KB	128k 256k 64k 128k 256k 512k 16k	On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash	MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core	
ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 556kB Flash CC2538xx 128 kB CC2538xx 256 KB CC2538xx 256 KB CC2538xx 256 KB CC2538xx 128 kB CC2538xx 128 kB CC2538xx 14 kB CC2538x 14 kB CC2538xx 14 kB CC2538x 14 kB CC2588x 14 kB CC2588x 14	128k 256k 64k 128k 256k 512k 16k 32k	On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash	MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core	
ATSAMR21 128kB Flash ATSAMR21 256kB Flash CC2538xx 128 KB CC2538xx 512 KB CC2538xx 512 KB CC2538xx 512 KB CC2538xx 512 KB CC2538xx 512 KB CC2530x 512 KB CC2530x Flash	128k 256k 64k 128k 256k 512k 16k 32k 256k	On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash	MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core	
ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 56kB Flash CC2538x 128 KB CC2538x 512 KB CC2538x 512 KB CC2538x 512 KB CY8C40x (16kB Flash CY8C40x (16kB Flash CY8C50x Flash CY8C50x Configuration	128k 256k 64k 128k 256k 512k 19k 256k 256k 32k	On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash On-chip Rash	MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core	
ATSAMR21 128kB Flash ATSAMR21 256kB Flash CC2538xx 128 KB CC2538xx 512 KB CC2538xx 512 KB CC2538xx 512 KB CC2538xx 512 KB CC98C42xx TMACRO (32KB) CY8C50xx C16 KD CY8C50xx CFG NVL CY8C50xx CFG NVL CY8C50xx EEPROM	128k 256k 64k 128k 256k 512k 19k 32k 256k 32k 4B	On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash	MDK Core MDK Core	^
ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 556kB Flash CC2538xx 128 KB CC2538xx 128 KB CC2538xx 256 KB CC2538xx 256 KB CC3538xx 128 KB CC3538xx 128 KB CC3538xx 128 KB CC3538xx 104 KB CC3538x 104 KB CC3588x 104 KB CC3588x 104 KB CC3588x 1	128k 256k 64k 128k 256k 513k 16k 32k 256k 32k 48 2k	On-chip Flash On-chip Flash	MDK Core MDK Core	^
ATSAMR21 128kB Flash ATSAMR21 256kB Flash CC2538x 128 KB CC2538x 256 KB CC2538x 256 KB CC2538x 256 KB CC2538x 256 KB CC3538x 512 KF CY8C42x 1MACRO (32KB) CY8C50x Flash CY8C50x Flash CY8C50x Flash CY8C50x Flash Protection CY8C50x Rash Protection CY8C50x WO NVL	128k 256k 64k 128k 256k 512k 19k 256k 32k 256k 32k 256B 4B	On-chip Rash On-chip Rash	MDK Core MDK Core	
ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 56kB Flash CC2538x 256 KB CC2538x 512 KB CC2538x 512 KB CC2538x 512 KB CY8C40x (16kB) Flash CY8C40x (16kB) Flash CY8C50x Flash CY8C50x Configuration CY8C50x CER NVL CY8C50x EEPROM CY8C50x Flash Protection	128k 256k 64k 128k 256k 512k 16k 	On-chip Flash On-chip Flash	MDK Core MDK Core	
ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 64kB Flash CC2538xx 128 KB CC2538xx 512 KB CC2538xx 512 KB CC2538xx 512 KB CC98242xx TMACRO (32kB) CY8C50xx CFG NVL CY8C50xx CFG NVL CY8C50xx EEPROM CY8C50xx EEPROM CY8C50xx EEPROM CY8C50xx WO NVL EFM32 Gecko/Tmx Gecko	128k 256k 64k 128k 256k 512k 19k 256k 32k 256k 32k 256B 4B	On-chip Flash On-chip Flash	MDK Core MDK Core	
ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 64kB Flash CC2538xx 128 KB CC2538xx 512 KB CC2538xx 512 KB CC2538xx 512 KB CC98242xx TMACRO (32kB) CY8C50xx CFG NVL CY8C50xx CFG NVL CY8C50xx EEPROM CY8C50xx EEPROM CY8C50xx EEPROM CY8C50xx WO NVL EFM32 Gecko/Tmx Gecko	128k 256k 64k 128k 256k 512k 16k 256k 32k 48 256k 32k 48 256B 48 128k	On-chip Flash On-chip Flash	MDK Core MDK Core	· ·

Figure 4-9. Flash Download Tab in Target Driver Setup

13. Select the **Debug** tab. Make sure that KitProg2 is listed in the CMSIS-DAP – JTAG/SW adapter. Select the port as SW; the SW Device field will show IDcode as 0x0BB11477. Set the clock frequency to 1MHz and the Reset option to VECTRESET as shown in Figure 4-10. Click OK and program the device using the option Flash > Download.

Figure 4-10. Debug Tab in Target Driver Setup

Cortex-M Target Driver Setup Debug Flash Download CMSIS-DAP - JTAG/SW Adapter KtProg2 CMSIS-DAP Serial No: 0B0C05B3031054	SW Device IDCODE Device Name SWDIO Ox0BC11477 ARM CoreSight SW-DP	Move Up Down
Firmware Version: 1.0	Automatic Detection ID CODE: Manual Configuration Device Name: Add Delete Update	AP: 0x00
Connect & Reset Options Connect: Normal ✔ Rese ✔ Reset after Connect	t VECTRESET ▼ I Cache Code	ad Options fy Code Download wnload to <u>F</u> lash
	OK Cancel	Help



14. For debugging, go to the **Debug** tab and select the CMSIS-DAP Debugger as shown in Figure 4-11.



Options for Target 'CapSense_LowPower'	
Device Target Output Listing User C/C++ Asm C Use §imulator Settings Imit Speed to Real-Time	✓ Use: CMSIS-DAP Debugger ✓ Settings Stellaris ICDI Signum Systems JTAGjet ✓ Load J-LINK / J-TRACE Cortex ULINK Pro Cortex Debugger ULINK Pro Cortex Debugger Initialization NULink Debugger Videbug Stabs UDA Debugger
Restore Debug Session Settings Image: Breakpoints Image: Toolbox Image: Breakpoints Image:	Restor Least Model Debugger
CPU DLL: Parameter: SARMCM3.DLL	Driver DLL: Parameter: SARMCM3.DLL
Dialog DLL: Parameter: DARMCM1.DLL pCM0+	Dialog DLL: Parameter: TARIMCM1.DLL PCM0+
ОК Са	ncel Defaults Help

15. Selecting **Debug > Start/Stop Debug Session** will start the debug session. Note that the green LED on the CY8CKIT-041-40XX blinks at an 8-Hz rate during the debug session.

egister	Value	0x0000FFEE 0000 MOVS r0,r0	
Core	1800	0x0000FFF0 0000 MOVS r0,r0	
RO	0x40100004	0x0000FFF2 0000 MOVS r0,r0	
R1	0x00000145	0x0000FFF4 0000 M0V5 r0,r0	
- R2	0x00000000	0x0000FFF6 0000 MOVS r0.r0	
-R3	0x00000000		
	0x20000200	startup_CM0.s	
R5	0x08000000	at initial as	
RS	0x0FFFF060	31initial_sp 32	
- R7	0x00000000	32	
- R8	0x00000830	34 : <h> Heap Configuration</h>	
R9	0x0008C000	35 : <o> Heap Size (in Bytes) <0x0-0xFFFFFFFF8></o>	
R10	0x81020142	36 :	
R11	0x12000808	37	
R12	0x30110452	38 Heap Size EQU 0x0000000	
R13 (SP)	0x20000200	39	
R14 (LR) R15 (PC)	0x10000120 0x00000144	40 AREA HEAP, NOINIT, READWRITE, ALIGN=3	
RTS (PC) R=xPSR	G+6100000	41 heap base	
Banked	045100000	42 Heap Men SPACE Heap Size	
System		43 heap limit	
internal		44	
Mode	Thread	45	
Privlege	Privleged	46 PRESERVES	
Sack	MSP	47 THUND	
		48	
		49	
		50 ; Vector Table Mapped to Address 0 at Reset	
		51	
		52 AREA RESET, DATA, READONLY	
		(
roject Regi	sters	Text Editor Configuration Waard	
mand		a 🖬	Call Stack + Locals

Figure 4-12. Debug Session in μ Vision

5. Mass Storage Programmer



The KitProg2 can act as a USB Mass Storage Programmer. CMSIS-DAP and Mass Storage modes are the same modes though the method of programming is different. This is an alternative configuration of KitProg2. The KitProg2 programmer and debugger (PPCOM), KitProg2 USB-I2C Bridge, KitProg2 USB-SPI Bridge, and KitProg2 USB-UART Bridge functionalities are not available in this configuration.

5.1 Enter or Exit the Mass Storage Programmer Mode

Follow these steps to enter or exit the Mass Storage Programmer mode of KitProg2:

- Connect the kit to the PC. Ensure that the amber status LED is ON and not blinking. See section 10.1 KitProg2 Status LED Indication on page 71 for details on the status LED indications. Amber status LED ON indicates that the current configuration is KitProg2 Programmer and Debugger.
- Press and release the mode switch within two seconds. The KitProg2 re-enumerates as a Mass Storage Programmer/CMSIS-DAP programmer if the previous configuration is PPCOM programmer and debugger. The amber status LED stops glowing indicating that the kit is in Mass Storage Programming/CMSIS-DAP programming mode.
- The KitProg2 remains in the selected mode until you change the mode manually using the mode switch. To exit the Mass Storage Programming mode, press and release the mode switch within two seconds. The amber status LED is continuously ON, indicating that the KitProg2 is in the KitProg2 Programmer and Debugger mode.

5.2 Programming Using the Mass Storage Programmer

Follow these steps to program the target device using the Mass Storage Programmer:

1. Enter the Mass Storage Programmer mode as explained in Enter or Exit the Mass Storage Programmer Mode on page 27. The KitProg2 is visible as a removable disk drive in the file explorer of the PC, as shown in Figure 5-1.

Figure 5-1. KitProg2 Emulated as Mass Storage Device





2. Open the KitProg2 drive to view the *STATUS.TXT* file, as shown in Figure 5-2. Note that the file extension *.TXT* is visible for the file only if it is enabled in your PC settings. The *STATUS.TXT* file shows the current status of the Mass Storage Programmer.

Figure 5-2. STATUS.TXT in the KitProg2 Drive

 Computer KitProg2 (E:) 		And the second second second	10.000	10.000
 Share v 	with 🔻 Burn New folder			
ites	Name	Date modified	Туре	Size
ktop vnloads	🗋 STATUS	5/22/2013 3:27 PM	Text Document	1 KB

3. Copy any PSoC 4000S device based project *.hex* file to the KitProg2 drive to begin programming. Alternatively, you can drag and drop the *.hex* file onto the drive. The *.hex* file for a PSoC Creator project is available in the following path:

```
<Project Directory>\<Project Name.cydsn>\CortexM0p\<Compiler Name and Version>\<Debug> or <Release>\<Project Name.hex>
```

Figure 5-3. Copy .hex File to KitProg2 Drive

 KitProg2 (E:) 			_	
Burn New folder				
Name	Date modified	Туре	Size	
STATUS	5/22/2013 3:27 PM	Text Document	1 KB	
CY8CKIT_041_CapSense	1/16/2016 4:27 PM	HEX File	71 KB	
🖘 Copyir	ng			
Сору				
	Debug (G:\Projects\Kit_Prog\P	ania (Dahua) da KiaD		
	vering items	roje\Debug) to kitr	rogz (E:)	
ЭM	ore details		ancel	

4. The green status LED on the kit blinks during the programming operation. It stays ON after the programming operation completes successfully and the KitProg2 drive automatically removes the copied file from the drive. Press **[F5]** in the file explorer to refresh the contents of the drive. This will display only the *STATUS.TXT* file in the KitProg2 drive.

Note: For prototyping kits the amber LED blinks during the programming in Mass Storage mode. For more information on switching of modes in prototyping kits, refer Switching between KitProg2 Modes on page 7.

5. Open the *STATUS.TXT* file to view the status of the programming operation, as shown in Figure 5-4.

Figure 5-4. Status Displayed in KitProg2 Drive after Programming

r 🕨 KitProg2 (E:)			
▼ Print Burn New folder			
Name	Date modified	Туре	Size
STATUS	5/22/2013 3:27 PM	Text Document	1 KB
STATUS - Notepad			
<u>File Edit Format View H</u> elp			
Express KitProg2 Mass Storag Click the mode switch buttor NOTE: Only PSOC4 Families ar STATUS: The HEX file was pro	re supported in this	s version.	



5.3 Frequently Asked Questions on KitProg2 Mass Storage Programmer

- What are the Cypress kits supported by the KitProg2 Mass Storage Programmer? The KitProg2 Mass Storage Programmer supports all the kits mentioned in Table 2-1 on page 10.
 Note: Newer kits which are not listed in the table can be explored in Cypress website http:// www.cypress.com.
- What are the operating systems supported by KitProg2 Mass Storage Programmer? The KitProg2 Mass Storage Programmer works on Microsoft Windows (XP or later) and Apple Mac operating systems (OSX or later). The KitProg2 Mass Storage Programmer is currently not supported on Linux operating systems.
- 3. What happens if I copy an incorrect .hex file to the KitProg2 drive?

If you copy a *.hex* file with invalid data (such as incorrect silicon ID and incorrect checksum), the KitProg2 Mass Storage Programmer attempts a programming operation and generates an error indicating which step of the programming operation has failed in the *STATUS.TXT* file.

If you copy a *.hex* file which corresponds to any other device, the KitProg2 Mass Storage Programmer does not attempt a programming operation and generates an error indicating that the copied file is not a valid *.hex* file in the *STATUS.TXT* file.

If you copy any file other than the ones specified above, and the file size does not exceed the KitProg2 drive size, the file will be visible in the KitProg2 drive until the drive is removed from the PC. Note that the file is not actually copied to the KitProg2 drive. Delete these files before attempting to program a new *.hex* file.

4. Why does my operating system display the "Disk Not Ejected Properly" pop-up after every programming operation in KitProg2 Mass Storage Programmer mode?

The KitProg2 Mass Storage Programmer temporarily ejects two seconds after the programming operation. This can also cause the file explorer window of the KitProg2 drive to close after programming operation in some operating systems.

5. Is it possible to program an external PSoC other than the one on the kit using the KitProg2 Mass Storage Programmer?

Yes. You need to remove several onboard zero-ohm resistors to disconnect the onboard target device. See the documentation for the kit that you are using for details.

6. Can I use *.hex* files generated by any other IDE other than PSoC Creator to program the PSoC 4000S using KitProg2 Mass Storage Programmer?

Yes. You can use the *.hex* file generated by an external IDE such as Eclipse, IAR, or Keil μ Vision which supports PSoC 4 devices, to program using the KitProg2 Mass Storage Programmer.

7. Why does the programming time for different files vary?

The KitProg2 Mass Storage Programmer intelligently programs only the flash rows with non-zero data. Depending on the contents of your project, the programming time may take up to 30 seconds.

6. USB-UART Bridge



The KitProg2 can act as a USB-UART Bridge. This feature of the KitProg2 is useful to send and receive data between the Cypress device on the kit and a PC. For example, in the PSoC 4 S-Series Pioneer Kit, the KitProg2 USB-UART can be used to print debug messages on COM terminal software running on the PC.

This section demonstrates a method to create a PSoC 4 code example, which communicates with COM terminal software using the KitProg2 USB-UART Bridge. This example uses Windows HyperTerminal as the COM terminal software. If you have a Windows operating system that does not have HyperTerminal, use an alternative terminal software such as PuTTY.

 Create a new PSoC 4000S project in PSoC Creator, as shown in Figure 6-1. Select a specific location for your project and name the project as desired. You must select the appropriate target hardware (kit) for this project. This example uses CY8CKIT-041-40XX (PSoC 4000S device) as the target hardware. Ensure that the **Select project template** option is set to 'Empty schematic'. This example uses PSoC 4000S as the target device and PSoC 4 S-Series Pioneer Kit as the target board.

Figure 6-1. Create New Project in PSoC Creator

Create Project - CY8CKI	-041-40XX (PSoC 4000S)		? ×
Select project type Choose the type of p	roject – design, library, or workspace.		
Design project:			
Target kit:	CY8CKIT-041-40XX (PSoC 4000S)		-
 Target module: Target device: Library project 	CY8CKIT-041-40XX (PSoC 4000S) CY8CKIT-042-BLE (PRoC BLE) CY8CKIT-042-BLE (PRoC BLE) CY8CKIT-042-BLE (PRoC BLE) CY8CKIT-042-BLE (PSoC 4200 BLE) CY8CKIT-044 (PSoC 4200M) CY8CKIT-048 (PSoC Analog Coprocessor)		
Workspace	CY8CKIT-145-40XX (PRoC BLE) CY8CKIT-145-40XX (PSoC 4000S) CY8CKIT-146 (PSoC 4200M) CY8CKIT-146 (PSoC 4200M) CY8CKIT-147 (PSoC 4100PS)		
		Next >	Cancel



Create Project - Kit: CY8CKIT-041 (PSoC 4000S)	? <mark>×</mark>
Select project template Choose a schematic template or start your design with a kit or example project.	
Code example Choose from our library of code examples.	
Pre-populated schematic Start with typical MCU functions (like UART, ADC, etc.).	
Empty schematic Create a full custom design by adding functionality from the component catalog.	
< <u>B</u> ack Next >	Cancel

Create Project Choose a name a	and location for your design.
Workspace:	Create new workspace
Workspace name:	CY8CKIT-041
Location:	C:\Users\PSoC\Project
Project name:	CY8CKIT_041_UART]
	< Back Einish Cancel



 Drag and drop a UART (SCB mode) Component from the Component Catalog (see Figure 6-2) to the TopDesign. The Component Catalog is located along the right of the PSoC Creator window by default. To configure the UART, double-click or right-click the UART Component and select Configure, as shown in Figure 6-3.

Figure 6-2. UART Component in Component Catalog



Figure 6-3. Open UART Configuration Window





3. Configure the UART Component as shown in Figure 6-4, Figure 6-5, and Figure 6-6, and then click **OK**.

Figure 6-4. UART Configuration Tab Window

2 X
4 Þ
Cancel

Figure 6-5. UART Basic Tab Window

Configure 'SCB_P4'	8	X
Name: UART_1		
Configuration	UART Basic UART Advanced Built-in	4 Þ
Mode:	Standard	<u>^</u>
Direction:	TX + RX 🔹	
Baud rate (bps):	9600 Actual baud rate (bps): 9615	
Data bits:	8 bits 💌	
Parity:	None	
Stop bits:	1 bit 💌	=
Oversampling:	12	
Clock from termina	al de la constante de la const	
Median filter		
Retry on NACK		
Inverting RX		
Enable wakeup fro	om Deep Sleep Mode	
Low power receive	ing	-
Datasheet	OK Apply Canc	el



Configuration UART Basic	JART Advanced Built-in	4 ا
Buffers size RX buffer size: 8	Interrupt DMA None RX out Internal TX out External	
Interrupt sources UART done TX FIFO not full TX FIFO empty TX FIFO overflow TX FIFO underflow TX FIFO underflow TX lost arbitration TX NACK TX FIFO level	 RX FIFO not empty RX FIFO full RX FIFO overflow RX FIFO underflow RX frame error RX parity error RX FIFO level 	
FIFO levels TX FIFO: 0	RX FIFO: 7 -	
Accept matching address in RX F Flow control RTS Polarity: Active Low CTS Polarity: Active Low	▼ RTS FIFO level: 4	< V

Figure 6-6. UART Advanced Tab Configuration Window



4. Select P0[4] for UART RX and P0[5] for UART TX in the **Pins** tab of *<Project_Name>.cydwr*, as shown in Figure 6-7. This can be opened by double clicking on "Pins" under the "Design Wide Resources". The *<Project_Name>.cydwr* file can be found in the Workspace Explorer window, which is located along the left of the PSoC Creator window. Note that these pins are for the USB-UART interface on the PSoC 4 S-Series Pioneer Kit. If you are using a different kit, refer to the respective kit guide for the appropriate pins.

Note: UART RX and UART TX can be routed to any digital pin on PSoC 4 by using the UDB implementation of the UART Component. Here, the SCB implementation of the UART is used, which routes the pins to one of the specific set of pins supported by the device. This will vary depending on the PSoC 4 device used.

Figure 6-7. UART Pin Assignment





5. Place the following code in the *main.c* file. The code echoes any data received through the UART.

Note: The *main.c* file can be found under Source Files on the Workspace Explorer window, which is located along the left of the PSoC Creator window by default. Double-click on the file to open it.

 Build the project by choosing Build > Build [Project Name] or pressing [Shift] [F6]. After the project is built without errors and warnings, program the project (by choosing Debug > Program) to the PSoC 4000S using KitProg2.

```
#include <project.h>
int main()
{
      uint8 ch;
      /* Start SCB UART TX+RX operation */
      UART 1 Start();
      /* Transmit String through UART TX Line */
      UART 1 UartPutString("CY8CKIT-041 USB-UART");
      for(;;)
      {
      /* Get received character or zero if nothing has been received yet */
             ch = UART 1 UartGetChar();
             if(0u != ch)
             {
              /* Send the data through UART. This function is blocking and waits
                       until there is an entry into the TX FIFO. */
                    UART 1 UartPutChar(ch);
             }
      }
```


To communicate with the PSoC 4000S device from the terminal software, follow this procedure:

1. Connect the USB Micro-B cable to J6. The kit enumerates as a **KitProg2 USB-UART**, and is available in the **Device Manager** under **Ports (COM & LPT)**. A communication port is assigned to the **KitProg2 USB-UART**, as shown in Figure 6-8.

Figure 6-8. KitProg2 USB-UART in Device Manager

🚔 Device Manager	x
<u>File</u> <u>Action</u> <u>View</u> <u>H</u> elp	
🗢 🔶 🔲 📓 🖬 🔯	
> 🍣 Batteries	*
Bluetooth Radios	_
🖂 🚛 Computer	
Disk drives	
🔉 📲 Display adapters	
DVD/CD-ROM drives	
🔈 🕼 Human Interface Devices	
De TOP ATA/ATAPI controllers	
Keyboards	
Mice and other pointing devices	
Monitors	
Network adapters	=
Other devices	
Ports (COM & LPT)	
ECP Printer Port (LPT1)	
KitProg2 USB-UART (COM43)	
Processors	
SD host adapters	
b - mail Smart card readers	
Sound, video and game controllers	
System devices	
D - Universal Serial Bus controllers	-

 Open HyperTerminal, choose File > New Connection, enter a name for the new connection, and then click OK as shown in Figure 6-9. For PuTTY, double-click the PuTTY application and select Serial under Category.

Figure 6-9. Open New Connection HyperTerminal

Connection Description	? ×
New Connection	
Enter a name and choose an icon for the connect	tion:
Name:	
USB-UART Comm	
Icon:	
•	÷.
ОК	Cancel

PuTTY

Reputry Configuration		×
Category: Session Logging Terminal Keyboard Bell Features	Options controlling Select a serial line Serial line to connect to Configure the serial line	COM43
- Window - Appearance - Behaviour - Translation - Selection - Colours - Connection - Data - Proxy - Teinet	Speed (baud) Data bits Stop bits Panty Flow control	9600 8 1 None • XON/XOFF •
About	5	Qoen Qancel



 A new window opens, where the communication port can be selected. In HyperTerminal, select COMx (the specific communication port that is assigned to the KitProg2 USB-UART) in Connect using and click OK, as shown in Figure 6-10.

In PuTTY, enter the COMx in Serial line to connect to. This example uses COM43.

Figure 6-10. Select Communication Port HyperTerminal

Ρ	u	T	T	Y	

	RuTTY Configuration		X
Connect To			
Connect To	Category: - Session - Logging - Terminal - Reyboard - Bell - Features - Window - Appearance - Behaviour - Translation - Selection - Colours - Connection - Data - Proxy - Tienet - Riogin - SSH - SSH - SSH - SSH	Options controlli Select a serial line Serial line to connect to Configure the serial line Speed (baud) Data bits Stop bits Parity Bow control	ng local serial lines COM43 9600 8 1 None XON/XOFF
	About		Open Cancel

 In HyperTerminal, select Bits per second, Data bits, Parity, Stop bits, and Flow control under Port Settings and click OK (see Figure 6-11). Ensure that the settings are identical to the UART settings configured for the PSoC 4000S device.

In PuTTY, enter the **Speed (baud)**, **Data bits**, **Stop bits**, **Parity**, and **Flow control** under **Configure the serial line**.

Figure 6-11. Configure the Communication Port HyperTerminal PuTTY

COM43 Properties	8 ×
Port Settings	
Bits per second:	9600 💌
<u>D</u> ata bits:	8 💌
Parity:	None 👻
Stop bits:	1 •
Eow control:	None
	Restore Defaults
	K Cancel Apply

PuTTY Configuration		×
Category:		
- Session	Options controlling	local serial lines
└─ Logging ⊡- Terminal └─ Keyboard	Select a serial line Serial line to connect to	COM43
Bell Features	Configure the serial line	
E- Window	Speed (baud)	9600
- Appearance - Behaviour	Data <u>b</u> its	8
Translation	Stop bits	1
- Selection - Colours	Parity	None 🔻
- Connection	Elow control	XON/XOFF -
- Proxy - Teinet - Riogin B- SSH - Senia		
About		Den Cancel



 Enable Echo typed characters locally under File > Properties > Settings > ASCII Setup to display the typed characters on HyperTerminal, as shown in Figure 6-12. In PuTTY, select Force on under Terminal > Line discipline options to display the typed characters on PuTTY, as shown in Figure 6-12.

Figure 6-12. Enable Echo of Typed Characters in HyperTerminal and PuTTY
HyperTerminal
PuTTY

ASCII Setup	RuTTY Configuration	×	
ASCII Sending Send line ends with line feeds Echo typed characters locally Line delay: milliseconds. Character delay: milliseconds. ASCII Receiving Assent line feed to incoming line and to the second line feed to incoming line and to the second line feed to incoming line and to the second line feed to incoming line and to the second line feed to incoming line and to the second line feed to incoming line and to the second line feed to incoming line and to the second line feed to incoming line and to the second line feed to be a second line second line second line feed to be a se	Category: - Session - Logging - Keyboard - Reyboard - Bell - Features - Window - Appearance - Behaviour - Translation - Selection - Colours - Colours	Options controlling the terminal emulation Set various terminal options Auto wrap mode initially on DEC Origin Mode initially on Implicit CB in every LF Implicit LE in every CR Use background colour to erase screen Egable blinking text Angwerback to ^E: PuTTY Line discipline options	
Append line feeds to incoming line ends Force incoming data to 7-bit ASCII V Mrap lines that exceed terminal width OK Cancel	Connectori Data Proxy Telnet Riogin ⊕ SSH Serial	Line discipline options Local echo: Auto Force on Force on Force off Local line editing: Auto Force on Force off Remote-controlled printing Printer to send ANSI printer output to: Open Cancel	

 In PuTTY, click Session and select Serial under Connection type. The Serial line shows the communication port (COM43) and Speed shows the baud rate selected. Click Open to start the communication, as shown in Figure 6-13.

Figure 6-13. Opening Port in PuTTY

🕵 PuTTY Configuration		×	
Category:			
E Session	Basic options for your PuTTY se	ession	
Logging	Specify the destination you want to conne		
- Keyboard	Serial line COM43	Speed 9600	
- Bell - Features Window	Connection type: Raw <u>I</u> elnet Rlogin <u>S</u> S		
- Appearance - Behaviour - Translation - Selection	tion Saved Sessions		
Colours Connection Data Proxy Telnet Rlogin PSSH	Default Settings	Load Sa <u>v</u> e Delete	
- Serial	Close window on exit: Always Never Only on c	lean exit	
About	Qpen	Cancel	



7. The COM terminal software displays both the typed data and the echoed data from the PSoC 4000S UART, as shown in Figure 6-14.

Note: The string "CY8CKIT-041 USB-UART" is transmitted when the kit is powered up or reset. If you open the terminal window after the kit has been plugged in, you will not see this message. Press the Reset button on the kit to see the message.

Figure 6-14. Data Displayed on HyperTerminal and PuTTY



7. USB-I2C Bridge



The KitProg2 serves as a USB-I2C Bridge that can be used to communicate with USB-I2C software running on a PC. For example, the KitProg2 USB-I2C Bridge can be used to tune the CapSense Component on a PSoC device. This feature is applicable to all kits listed in Table 2-1 on page 10. This section uses the PSoC 4 S-Series Pioneer Kit as an example to demonstrate the KitProg2 USB-I2C Bridge functionality. The following steps describe how to use the USB-I2C Bridge, which can communicate between the Bridge Control Panel (BCP) software and the PSoC 4000S device.

Note: For information on how to use the KitProg2 USB-I2C Bridge to tune the CapSense Component, refer to the Manual Tuning Process section in AN85951 - PSoC 4 CapSense Design Guide.

 Create a new PSoC 4000S project in PSoC Creator. Select a specific location for your project and name the project as desired. You must select the appropriate target hardware (kit) for this project as shown in Figure 7-1. This example uses CY8CKIT-041-40XX (PSoC 4000S device) as the target hardware. Ensure that the **Select project template** option is set to 'Empty schematic' as shown in Figure 7-2. Create the workspace and project name as shown in Figure 7-3.

Figure 7-1. Create New Project in PSoC Creator

Create Project - CY8CKI	-041-40XX (PSoC 4000S)	? ×
Select project type Choose the type of p	roject – design, library, or workspace.	
Design project:	CY8CKIT-041-40XX (PSoC 4000S) CY8CKIT-041-40XX (PSoC 4000S) CY8CKIT-042-BLE (PRoC BLE)	
Target device:	CY8CKIT-042-BLE (PRoC BLE) CY8CKIT-042-BLE (PRoC BLE) CY8CKIT-042-BLE (PSoC 4200 BLE)	
Library project	CY8CKIT-044 (PSoC 4200 BLE) CY8CKIT-044 (PSoC 4200M) CY8CKIT-048 (PSoC Analog Coprocessor)	
Workspace	CY8CKIT-145-40XX (PRoC BLE) CY8CKIT-145-40XX (PSoC 4000S) CY8CKIT-145 (PSoC 4200M) CY8CKIT-146 (PSoC 4200M) CY8CKIT-147 (PSoC 4100PS)	
		Next > Cancel



Figure 7-2. Select Empty Schematic

Create Project - Kit: CY8CKIT-041 (PSoC 4000S)	? <mark>x</mark>
Select project template Choose a schematic template or start your design with a kit or example project.	
Code example Choose from our library of code examples.	
Pre-populated schematic Start with typical MCU functions (like UART, ADC, etc.).	
Empty schematic Create a full custom design by adding functionality from the component catalog.	
< Back Next >	Cancel

Figure 7-3. Create Workspace

Create Project - Kit: C	Y8CKIT-041 (PSoC 4000S)	? ×
Create Project Choose a name a	nd location for your design.	
Workspace:	Create new workspace	•
Workspace name:	CY8CKIT-041	
Location:	C:\Users\PSoC\Project	
Project name:	CY8CKIT_041_I2C	
	< Back Einish	Cancel



 Drag and drop an EZI2C Slave (SCB mode) Component from the Component Catalog (see Figure 7-4) to the TopDesign. The Component Catalog is located along the right of the PSoC Creator window by default. To configure the EZI2C Slave Component, double-click or right-click the EZI2C Slave Component and select Configure, as shown in Figure 7-5.

Figure 7-4. EZI2C Slave Component in Component Catalog



Figure 7-5. Open EZI2C Slave Configuration Window





3. Configure the EZI2C Slave Component as shown in Figure 7-6 and Figure 7-7; then, click **OK**.

Figure 7-6. Configuration Tab

0 C 1000 D.4	<u> </u>	? ×
Configure 'SCB_P4'		
Name: 57/20-1		
Name: EZI2C_1		
Configuration	EZI2C Basic EZI2C Advanced Built-in	4 Þ
Unconfigured SCB		
I2C		
EZI2C		
SPI		
O UART		
Datasheet	OK Apply	Cancel



Configuration EZI2C Basic EZI2C Advanced Built-in ata rate (kbps): ID Actual data rate (kbps): 100 Clock from terminal Clock stretching Dyte mode Image: Stretching Image: Byte mode Image: Stretching image: Stretching Image: Stretching	Configuration EZI2C Basic EZI2C Advanced Built-in ata rate (kbps): 100 Actual data rate (kbps): 100 Clock from terminal Clock stretching Byte mode umber of addresses: 1 imary slave address (7-bits): 0x08	nfigure 'SCB_P4'		Ş
lata rate (kbps): IOI Actual data rate (kbps): 100 Clock from teminal Clock stretching Byte mode Image: Stretching lumber of addresses: 1 rimary slave address (7-bits): 0x08 secondary slave address (7-bits): 0x09	ata rate (kbps): 100 - Actual data rate (kbps): 100 Clock from terminal Clock stretching Byte mode umber of addresses: 1 - imary slave address (7-bits): 0x08 econdary slave address (7-bits): 0x09 ub-address size (bits): 8 -	Name: EZI2C_1		
ata rate (kbps): IO Actual data rate (kbps): 100 Clock from terminal Clock stretching Clock stretching Byte mode lumber of addresses: 1 trimary slave address (7-bits): 0x08 iecondary slave address (7-bits): 0x09	ata rate (kbps): 100 - Actual data rate (kbps): 100 Clock from terminal Clock stretching Byte mode umber of addresses: 1 - imary slave address (7-bits): 0x08 econdary slave address (7-bits): 0x09 ub-address size (bits): 8 -	Configuration EZI2C Basic	EZI2C Advanced Built-in	
Clock stretching Byte mode lumber of addresses: 1 rimary slave address (7-bits): lecondary slave address (7-bits):	Clock stretching Byte mode umber of addresses: imary slave address (7-bits): 0x08 econdary slave address (7-bits): 0x09 ub-address size (bits): 8 ▼		Actual data rate (kbps): 100	
Byte mode lumber of addresses: 1 rimary slave address (7-bits): 0x08 iecondary slave address (7-bits):	Byte mode umber of addresses: imary slave address (7-bits): 0x08 econdary slave address (7-bits): 0x09 ub-address size (bits): 8	Clock from terminal		
Imper of addresses: I trimary slave address (7-bits): 0x08 iecondary slave address (7-bits): 0x09	umber of addresses: 1 imary slave address (7-bits): 0x08 econdary slave address (7-bits): 0x09 ub-address size (bits): 8	Clock stretching		
trimary slave address (7-bits): 0x08 iecondary slave address (7-bits): 0x09	imary slave address (7-bits): 0x08 econdary slave address (7-bits): 0x09 ub-address size (bits): 8	Byte mode		
econdary slave address (7-bits):	acondary slave address (7-bits): 0x09 ub-address size (bits): 8	Number of addresses:	1 •	
	ub-address size (bits):	Primary slave address (7-bits):	0x08	
		Secondary slave address (7-bits):	0x09	
ub-address size (bits): 8 -	Enable wakeup from Deep Sleep Mode	Sub-address size (bits):	8 -	
Enable wakeup from Deep Sleep Mode		Enable wakeup from Deep Sleep	Mode	

Configure 'SCB_P4'	? <mark>×</mark>
Name: EZI2C_1	
Configuration EZI2C Basic EZI2C Advanced Built-in	4 ۵
Slew rate: Fast	
I2C bus voltage (V): 3.3	
Datasheet OK Apply	Cancel
	i

ОК

Apply

Cancel

Datasheet



4. Select pin P3[0] for the I2C SCL and pin P3[1] for the I2C SDA in the **Pins** tab of <<u>Project_Name>.cydwr</u>, as shown in Figure 7-8. This can be opened by double clicking on "Pins" under the "Design Wide Resources". The <<u>Project_Name>.cydwr</u> file can be found in the Work-space Explorer window, which is located along the left of the PSoC Creator window. Note that these are the pins for the USB-I2C interface on the PSoC 4 S-Series Pioneer Kit. If you are using a different kit, refer to the respective kit guide for the appropriate pins.

Figure 7-8. Select Pins in cydwr





5. Place the following code in the *main.c* file. The code will enable the PSoC 4000S device with the BCP application using the EZI2C Slave interface.

Note: The *main.c* file can be found on the Workspace Explorer window, which is located along the left of the PSoC Creator window by default. Double-click on the file to open it.

```
#include <project.h>
#define BUF SIZE
                                  0x0A
#define READ WRITE SIZE
                                  0x05
int main()
{
      /* I2C Read/Write Buffer. */
      uint8 i2cBuffer[BUF SIZE] = {0x01, 0x02, 0x03, 0x04, 0x05,
                                    0x0A, 0x0B, 0x0C, 0x0D, 0x0E};
      CyGlobalIntEnable;
      EZI2C 1 Start();
      /* This API sets the buffer and address boundary to which the external
       * master can communicate. In this example, external master can read
       * from and write to the first 5 bytes of the i2cBuffer and read bytes
       * from all the 10 bytes of the i2cBuffer array. */
      EZI2C 1 EzI2CSetBuffer1(BUF SIZE, READ WRITE SIZE, i2cBuffer);
      for(;;)
      {
      }
```

- Build the project by choosing Build > Build Project or pressing [Shift] [F6]. After the project is built without errors and warnings, program ([Ctrl] [F5]) this project onto the PSoC 4000S using KitProg.
- Open BCP from Start > All Programs > Cypress > Bridge Control Panel <version> > Bridge Control Panel <version>.



- 8. Select **KitProg2/<serial number>** under **Connected I2C/SPI/RX8 Ports**, as shown in Figure 7-9.
- 9. If the KitProg2 firmware is not the most recent version the connection will not work. See Updating the KitProg2 Firmware on page 17 to update the KitProg2 firmware.

Figure 7-9. Connecting to KitProg2 in BCP

🗱 Bridge Control Panel	
<u>File Editor Chart Execute Tools Help</u>	
● ■ 単 圖 ● ■ ■ ▼ ■ ▼ ■ ■ ■	
Edtor Chart Table File	
	^
	Ŧ
Consider Dest	•
Opening Port Successfully Connected to KitProg2/0D14193D01344400	^
KitProg2 Version 0.06 [HW Rev.0x01]	
This window shows status of performed commands and reseived data	
٩	Þ
Consected ICC/SPL/RX2 Pote:	Protocol
Reset Send all strings: COM37	I2C
Stop Pepeat To file Connected and Ala	RX8 (UART)
Scan penod, ms: U	
1:1 Syntax: OK Connected Powered Voltage: 1966 mV	:



10.Open **Protocol Configuration** from the Tools menu and select the appropriate **I2C Speed**, as shown in Figure 7-10. Ensure that the I2C speed is the same as the one configured in the EZI2C Slave Component. Click **OK** to close the window.

Figure 7-10. Opening Protocol Configuration Window in BCP

🧱 Brid	ge Contro	l Panel						
File	Editor	Chart	Execute	Tools	Help			
🚅 🖬	.) h @		Pr	otocol Confi	guration	F7	
Editor	Chart 1	Table Fil	e	12	C Bootloader		F3	
								_

Protocol Configuration	
SPI I2C RX8 (UAF	(RT)
I2C Speed	
🔘 1 MHz 🔘 400 kHz	z 💿 100 kHz 💿 50 kHz
	OK Cancel



11. The buffer in the EZI2C slave is initialized with the data 0x01, 0x02, 0x03, 0x04, and 0x05 after power-on or reset. Send the Read command from the BCP to read the initial data with the slave address 0x08 (first command in Figure 7-11). The EZI2C Slave requires an additional write to be sent from the BCP to set the offset address from/to where the data bytes are read/written. Use a write command to set the read offset and then issue the read command. In this example, the offset is set to '0' so the five data bytes are read starting from the beginning of the buffer. The log shows the initial data. After reading, transfer five bytes of data from the BCP to the I2C device with slave address 0x08. The five bytes sent are 0x11, 0x22, 0x33, 0x44, and 0xAA as shown in the second command of Figure 7-11. Type the command shown in Figure 7-11 and press [Enter] or click the Send button in the BCP. The log shows whether the transaction was successful. A "+" after a byte indicates that the transaction was successful, and a "--" indicates that the transaction failed.

Note: You can add additional lines of commands by pressing **[Ctrl] [Enter]**. To execute any line, click on that line and press **[Enter]** or click the **Send** button.

Figure 7-11. Enter Commands in BCP

🗱 Bridge Control Panel
Eile Editor Chart Execute Iools Help
◎ ■ ※ ◎ ◎ ◎ ● ○ Ⅲ 〒 〒 22
Editor Chart Table File
w 8 0 r 8 x x x x p
w 8 0 11 22 33 44 AA p
Generate STOP condition on I2C bus
Offset position
Slave Address
"Write Data" command
Indicates Acknowledgement (ACK)
x
Opening Port
Successfully Connected to KitProg2/0B0C05B303105400 KitProg2 Version 0.08 [HW Rev.0x01] Set the offset pointer to 0 and read 5 bytes
w 08+ 00+ r 08+ 01+ 02+ 03+ 04+ 05+ p
W 08+ 00+ 11+ 22+ 33+ 44+ AA+ p Set the offset pointer to 0 and write 5 bytes to the slave
Connected I2C/SPI/RX8 Ports:
Connector IzC/SP/VAS roits: Power Protocol KAProg2/UB0C058303105400 +5.0V Income Protocol COM43
■Stop Pepeat the Repeat count: 0
Scan period, ms: 0 (
3:23 Syntax:OK ok Connected Powered Voltage: 3367 mV



12. From the BCP, read back the five bytes of data just written from the I2C slave device with slave address 0x08. The log shows if the transaction was successful, as shown in Figure 7-12.

Figure 7-12. Read Data Bytes from BCP

🗱 Bridge Control Panel	
Eile Editor Chart Execute Iools Help	
● ■ ● ◎ ◎ ◎ ◎ ● ■ ▼ ■ ▼ ■	
Editor Chart Table File	
W 8 0 r 8 x x x x p Generate STOP condition on I2C bus No. of data bytes o be read Slave Address "Read Data" command Set Offset position Slave Address "Write Data" command Data bytes returned after read operation	*
é la	
Opening Port	*
Successfully Connected to KitProg2/0B0C05B303105400	
KitProg2 Version 0.08 [HW Rev.0x01] w 08+ 00+ r 08+ 01+ 02+ 03+ 04+ 05+ p	E
w 08+ 00+ 11+ 22+ 33+ 44+ AA+ p	
w 08+ 00+ r 08+ 11+ 22+ 33+ 44+ AA+ p	-
<	Þ
Image: Scope State Send all strings: Send all strings: Send all strings: Scaperiod, ms: O Connected I2C/SPI/RX8 Ports: Power Power +5.0V Image: Scope String Scan period, ms: O O COM43 Image: Scope String Power Image: Scope String Scan period, ms: O O Image: Scope String Image: Scop	Protocol IZC SPI RX8 (UART)
1:22 Syntax: OK ok Connected Powered Voltage: 3367 mV	

Refer to **Help > Help Contents** in the BCP or press **[F1]** for more information on the I2C commands. Refer to the EZI2C component datasheet for more information on the EZI2C protocol.

8. USB-SPI Bridge



The KitProg2 serves as a USB-SPI bridge that can be used to communicate with USB-SPI software running on a PC. This section uses the PSoC 6 BLE Kit as an example to demonstrate the KitProg2 USB-SPI Bridge functionality. The following steps describe how to use the USB-SPI bridge, which can communicate between the Bridge Control Panel (BCP) software and the PSoC 6 BLE device.

 Create a new PSoC 6 BLE project in PSoC Creator. Select a specific location for your project and name the project as desired. You must select the appropriate target hardware (kit) for this project as shown in Figure 8-1. This example uses CY8CKIT-062 kit (PSoC 6 BLE device) as the target hardware. Ensure that the **Select project template** option is set to 'Empty schematic' as shown in Figure 8-2. Create the workspace and project name as shown in Figure 8-3.

Create Project - CY8CKI	-062-BLE (PSoC 6 BLE)		? X
Select project type Choose the type of p	roject - design, library, or workspace.		
Design project:	CY8CKIT-062-BLE (PSoC 6 BLE)		•
 Target module: Target device: 			
 Target device: Library project 			
Workspace			
		Net	Canad
		Next >	Cancel

Figure 8-1. Create New Project in PSoC Creator



Figure 8-2. Select Empty Schematic

Create Project - CY8C637BZI-BLD74			8 ×
Select project template Choose a schematic template or start your de	esign with a kit or example project.		
Code example Choose from our library of code example	cs.		
Empty schematic Create a full custom design by adding f	unctionality from the component cat	alog.	
	< Back	Next >	Cancel

Figure 8-3. Create Workspace

Create Project - CY8C	637BZI-BLD74	?	×	J
Create Project Choose a name a	nd location for your design.			
Workspace:	Create new workspace	•		
Workspace name:	PSoC_6_SCB		_	
Location:	C:\Users\PSoC\Project			
Project name:	P6_SPI			
	< Back Finish	Canc	el	



- 2. Drag and drop an SPI component and digital output pin from the component catalog to the Top-Design. The component catalog is located along the right of the PSoC creator window by default.
- 3. Double-click the digital output pin to configure the pin. Set the pin name as **Pin_LED** and uncheck the hardware connection as shown in Figure 8-4.

nfigure 'GPIO_PDL' Name: Pin_LED Pins Built-In Number of pins: 1	Display as bus 🗙 🔊 🛊 🔹	2 - X
[All pins] └─⊠ [Rin_LED_0]	General Input Output Type Analog Ditve mode Strong Ditve Ditve mode Strong Ditve Ditve mode Strong Ditve Cutput HW connection Output enable Bidirectional External terminal	Initial drive state: Low (0) Min. supply voltage: Max frequency: 100 MHz Hot swap
•	1 m	,
Datasheet	ОК	Apply Cancel

Figure 8-4. Configuring the output pin

4. To configure the SPI component, double-click or right-click the **SPI Component** and select **Configure**, as shown in Figure 8-5.

Figure 8-5. SPI Component in Component Catalog and configuring SPI Component





5. Configure the SPI Slave Component as shown in Figure 8-6 then click **OK**.



Basic Advanced Pins I Sub Mode	Built-in Motorola		- 24	_	4
			 f(x) 	^	
SCLK Mode	_	CPOL = 0			
Data Rate (kbps)	1000		f(x)	_	
Enable Input Glitch Filter			f(x)		
Enable Wakeup from DeepSlee	p 🖾		f(x)		
Data Configuration					
Bit Order	MSB First		• f(x)		
RX Data Width	8		f(x)		
TX Data Width	8		f(x)	-	
ss					Actual data rate (kbps): 1066.098 (
SCLK					
MOSI D7 D6	D5 1	D4 D3	D2	c	
MISO D7 D6	05	04 03	D2	0	

6. Select pin P12[0] for SPI_MOSI, P12[1] for SPI_MISO, P12[2] for SPI_SCLK, P12[4] for SPI_SS and P0[3] for the digital output pin Pin_LED in the **Pins** tab of <Project_Name>.cydwr, as shown in Figure 8-7. The <*Project_Name>.cydwr* file is available in the Workspace Explorer window, which is located along the left of the PSoC Creator window by default. Double-click on the file to open it. Note that these are the pins for the USB-SPI interface on the PSoC 6 BLE Pioneer Kit. If you are using a different kit, refer to the respective kit guide for the appropriate pins.

Figure 8-7. Select Pins for SPI interface and the output pin

art Page	TopDes			P6_	SPLC		<u> </u>	0	0						•
	10	9	8	/	6	5	4	3	2	1	Name /	Port		Pin	
		-	PQ 3	-11	P118	e113	-	-43	-						
		-			-		(***	-	-	1000	\SPI_P6:miso_s\	P12[1]	•	B4	•
	Ā	Ā	-	-	ma	$\overline{\frown}$	Ă	-		-	\SPI_P6:mosi_s\	P12[0]	•	A4	•
	\mathbb{Z}	9	9	9	9	Q	Q	2	g		\SPI_P6:sclk_s\	P12[2]	•	C4	•
	(19)	(M 1)	-	(**	्यम	<u></u>	শ্ৰ	्ल्या	म्पन	•	\SPI_P6:ss_s\	P12[4]	•	C5	•
		-	-	-			-122	(***			Pin_LED	P0[3]	•	E3	•
	(43)	(m)	(11)	(ma		(ma)	(ear	-	-	veen					
		m	(***	m	-		-	-	1						
	-11		-12	-	-	-10	-	•		•					
	-13	-	-	-	-	-13	-13	-14		-					
		-	-	-	-	-	-	-	-	-					
	•	-	-11	*	-										
	-	-		-	•••	-	•			-					
				1	16-B0	BZI-B 3A-BL n viev	E								



7. Place the following code in the main_cm0p.c file. The code will enable the PSoC 6 BLE device to communicate with the BCP application using the SPI Slave interface. The function of the code is to control LED ON/OFF (LED 5) on the CY8CKIT-062 kit based on the command received from the BCP. A command word of 0x00 turns OFF the LED and 0x01 turns ON the LED.

Note: The *main_cm0p.c* file can be found on the Workspace Explorer window, which is located along the left of the PSoC Creator window by default. Double click on the file to open it.

```
#include <project.h>
#define BUFFER SIZE
                                  1
                                  0
#define LED ON
#define LED OFF
                                  1
int main(void)
{
       __enable_irq();/* Enable global interrupts. */
       /* Enable CM4. CY CORTEX M4 APPL ADDR must be updated if CM4 memory
layout is changed. */
       Cy SysEnableCM4(CY CORTEX M4 APPL ADDR);
      int8 RxBuffer[BUFFER SIZE];
       RxBuffer[0]=0;
       /* Place your initialization/startup code here (e.g. MyInst Start()) */
      SPI P6 Start();
      for (;;)
       {
             while(SPI P6 IsBusBusy());
             SPI_P6_ReadArray((void*)RxBuffer,BUFFER_SIZE);
             SPI P6 ClearRxFifo();
             switch(RxBuffer[0])
              {
                    case 0x00:
                    Cy GPIO Write (Pin LED 0, LED OFF); // Turn the LED OFF
                    break;
                    case 0x01:
                    Cy GPIO Write (Pin LED 0, LED ON); // Turn the LED ON
                    break;
                    default:
                    break;
             }
       }
```



- Build the project by choosing Build > Build Project or pressing [Shift][F6]. After the project is built without errors and warnings, program ([Ctrl][F5]) this project onto the PSoC 6 BLE using KitProg.
- 9. Open BCP from Start > All Programs > Cypress > Bridge Control Panel <version> > Bridge Control Panel <version>.
- Select KitProg2/<serial number> under Connected I2C/SPI/RX8 Ports, as shown in Figure 8-8. Select the protocol configuration and type in the command as shown in Figure 8-8. Refer BCP help topic for SPI command formats. Click on the "Send" radio button to send the command from BCP.

Findge Control Panel	-	- 0 ×
File Editor Chart Execute Tools Help	Protocol Configuration	
Edtor Chart Table File LC BOOTROBARY F3	SPI I2C RX8 (UART)	-
spi 01:x SPI Command	Shift Direction Slave Device: MSB Fint LSB Fint SS0 - P15[3] ▼ Mode Mode 00 Mode 01 Frequency: Mode 02 Mode 03 Hz	*
	OK Cancel	*
Select Port in the PortList, then try to Opening Port Successfully Connected to COM64 COM64 Serial Port Opening Port Successfully Connected to KitProg2/160709 KitProg2 Version 1.03 [HW Rev.0x02]		* *
	ted I2C/SPI/RX8 Pots: 2/150705E001155400	Protocol DI2C SPI RX8.00 RT)
2 : 1 Syntax : OK Co	onnected Powered Voltage: 3307 mV	.:

Figure 8-8. Connecting to Kitprog2 USB-SPI Bridge in BCP

Expected Output: Sending a command word of 0x01, turns LED5 ON and a command word of 0x00 turns LED5 OFF.

9. Developing Applications for PSoC 5LP



The KitProg2 is implemented using a PSoC 5LP device. You can also use the PSoC 5LP as a mixed-signal system-on-chip device to build your own custom projects. For example, the PSoC 5LP on the kit can be reprogrammed to act as a function generator for the kit. Refer to the application note AN69133 – PSoC 3 / PSoC 5LP: Easy Waveform Generation with the WaveDAC8 Component for details on how to create waveforms using a PSoC 5LP device.

Two types of projects can be created for a PSoC 5LP that runs KitProg2: Custom (Bootloadable) Application and Normal. Custom application projects can be bootloaded into the PSoC 5LP using the USB connection from a PC without any specialized hardware. To program normal projects, you will require a MiniProg3. You also need to populate the PSoC 5LP programming header on the development kit. For the PSoC 4 S-Series Pioneer Kit, this header is marked J11. See the respective kit guide for more information on the PSoC 5LP programming header. Go to section 9.2 Building a Normal Project for PSoC 5LP on page 67 for details on creating a normal project for PSoC 5LP.

Custom applications for KitProg2 are developed using the concept of dual image bootloading. To learn more about the bootloading concept, refer to the application note AN73854 – PSoC 3, PSoC 4, and PSoC 5LP: Introduction to Bootloaders.

The following sections provide step by step directions to build a Custom Application and a Normal project for KitProg2.

9.1 Building a Custom Bootloadable Application for KitProg2

Custom applications developed for the PSoC 5LP are programmed to the kit using the KitProg2 bootloader. Therefore, a bootloadable project should first be created. An example of a custom application is provided as a compressed archive *KitProg2_Custom_App.zip* in the PSoC Programmer installation folder under Examples\Misc\KitProg2_Custom_App\. This example demonstrates an LED with a breathing effect and also has the feature to switch between KitProg2 and custom application using the mode switch.



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

 In PSoC Creator, choose File > New > Project and select Target device; select <Launch Device Selector...> from the drop-down list as shown in Figure 9-1

Note: The custom application must provide a means to switch back to KitProg2 mode, otherwise the programming capability will be lost unless KitProg2 firmware is re-bootoladed.

Figure 9-1. Open New Project in PSoC Creator

Create Project - CY8C5888AXQ-LPC	96	? ×
Select project type Choose the type of project - desi	gn, library, or workspace.	
Design project: Target kit: Target module: Target device: PSoC 5LF Library project Workspace	 Last used: CY8C5888AXQ-LP096 Last used: CY8C5888AXQ-LP096 CY8C52LP CY8C56LP CY8C56LP CY8C58LP <launch device="" selector=""></launch> 	
	Next >	Cancel

2. Select CY8C5868LTI-LP039, as shown in Figure 9-2. Click OK and click Next.

Note: In PSoC Creator 3.1 or earlier, you must either set the Application Type as Bootloadable in the New Project window under the Advanced section, or you can change it after project creation by selecting Project > Build Settings and clicking <Project Name> > Application Type > Bootloadable. Beginning with PSoC Creator 3.2, the Application Type option is removed from the New Project window and the Build Settings menu. PSoC Creator 3.2 and later versions automatically recognizes the application type from the TopDesign schematic.

Figure 9-2. Select Device in PSoC Creator





3. Choose **Empty schematic** in the **Select project template** dialog, as shown in Figure 9-3. Click **Next**.

Figure 9-3. Select Empty Schematic

Create Project - CY8C5868LTI-LP039	? 💌
Select project template Choose a schematic template or start your design with a kit or example project.	
Code example Choose from our library of code examples.	
Empty schematic Create a full custom design by adding functionality from the component catalog.	
< <u>Back</u> <u>N</u> ext >	Cancel

4. In the **Create Project** dialog, choose the workspace name, location, and project name (Figure 9-4). Click **Finish**.

Figure 9-4. Choose Name and Location

eate Project - CY8	C5868LTI-LP039			? 💌
Create Project Choose a name a	and location for your design.			
Workspace:	Create new workspace			•
Workspace name:	KitProg2_Custom			
Location:	C:\KitProg2_Custom			
Project name:	KttProg2_Custom			
		< <u>B</u> ack	Einish	Cancel



5. Navigate to the Schematic view and drag and drop a Bootloadable Component (see Figure 9-5) on the TopDesign.

Figure 9-5.	Bootloadable	Component in	Component	Catalog
-------------	--------------	--------------	-----------	---------

Component Catalog (341 component	ents) 👻 🕈 🗙
Search for 🐴 🚺 🙆	
Cypress Default Off-Chip	4 4
Cypress Component Catalog	~
🗈 🚳 Analog	
Analog Primitives	
CapSense	
Communications	
Deprecated	
🖲 🚳 Digital	
🗈 🔯 Display	
🖲 🚳 External	
E M Filters	
E M Fixed Function	
Berry Ports and Pins	
Representation	
e e e e e e e e e e e e e e e e e e e	
Boost Converter [v5.0]	
Bootloadable [v1.50]	
Bootloader [v1.50]	
 Occk [v2.20] 	
- o cy_boot [v5.40]	
- o cy_dmac [v1.10]	
- O CyLFCIk [v1.10]	
- Die Temperature [v1.0]	
O Die Temperature [v2.0]	
- OMA [v1.70]	
- DMA Channel [v1.0]	
EEPROM [v3.0]	
— EMIF Port [v1.0]	
- Emulated EEPROM [v1.]	10]
🖲 🞯 External Memory Interfa	ce
-B Global Signal Reference	[v2.0]
 Global Signal Reference 	[v2.0] (2)
3 ILO Trim [v2.0]	
- ILO Trim [v2.0] (2)	
 Interrupt [v1.70] 	
 LIN_Dynamic [v3.40] 	
 Real-time clock (RTC) [v 	1.0]
- • RTC [v2.0]	
-SleepTimer [v3.20]	
SleepTimer [v3.20] (2)	
🖲 🚳 Thermal Management	•
Open datasheet Provides bootloadable application func	tionally.
Inst	
Bootloadab	le

6. All custom applications developed for the PSoC 5LP should be based on the KitProg2 bootloader *.hex* file, which is programmed onto the kit. Therefore, you need to provide the location of the bootloader *.hex* file inside the custom application project.

The bootloader .*hex* file is included in the compressed archive *KitProg2_Custom_App.zip* in the PSoC Programmer installation folder under <code>Examples\Misc\KitProg2_Custom_App\</code>. They are also present in Kit installation directory at <Install_Directory>\<Kit_Name>\<version>\Firmware\Programmer\KitProg2_Bootloader\.

Extract the Bootloader folder from the archive.



 Set the dependency of the Bootloadable Component by selecting the Dependencies tab in the configuration window and clicking the Browse button, as shown in Figure 9-6. Select the *KitProg2_Bootloader.hex* (see Figure 9-7) extracted in step 6 and click Open.

Note: The user should copy the *.hex* and *.elf* files into their custom project folder and refer to them from that location. The *KitProg2_Bootloader.elf* is selected automatically if it is available with the same name in the same path. Ensure that both *.hex* and *.elf* files exist in the same folder with the same name.

Figure 9-6. Configuration Window of Bootloadable Component

Configure 'Bootloadable'
Name: Bootloadable
General Dependencies Built-in 4 b
Bootloadable projects require a reference to the associated Bootloader project's HEX and ELF files. The HEX files extension is "hex. The ELF files extension depends on IDE and can be ".elf, ".out, ".axf, or other.
Bootloader HEX file:
C:\Users\Documents\PSoC Creator\KitProg2_Custom_App\Bootloader hex
Bootloader ELF file:
C:\Users\Documents\PSoC Creator\KitProg2_Custom_App\Bootloader.elf
Browse
Datasheet OK Apply Cancel

Figure 9-7. Select KitProg2 Bootloader Hex File.

Select a Bootloader Hex File		
KitProg2	_Custom_App > Bootloader - 47 Se	arch Bootloader 🔎
Organize 👻 New fold	ler	# • 🔟 🔞
🛛 🔆 Favorites	Name	Date modified Type
Libraries	KitProg2_Bootloader.hex	1/14/2016 4:00 PM HEX File
🛛 🖳 Computer		
Þ 🗣 Network		
		,
File <u>r</u>		x Files (*.hex) ▼ Open ▼ Cancel



8. In the General tab, check the Manual application image placement checkbox and set the Placement address as '0x00003200', as shown in Figure 9-8.

	Figure 9-8.	Bootloadable	Component -	General	Tab
--	-------------	--------------	-------------	---------	-----

Configure 'Bootloadable'				?	×
Name: Bootloadable					
General Dependencies Built	in				4 ۵
Application version: Application ID: Application custom ID: Manual application image placement					
Placement address:	0x00003200				
Checksum exclude section size (bytes):	0				
Datasheet		ОК	Apply	Cancel	

9. Develop your custom project. An example of a custom application is provided as a compressed archive *KitProg2_Custom_App.zip* in the PSoC Programmer installation folder under Examples\Misc\KitProg2_Custom_App\. This example demonstrates an LED with a breathing effect and also has the feature to switch between KitProg2 and custom application using the mode switch.



 Ensure that the <project name>.cydwr System setting of the bootloadable project and the KitProg2_Bootloader project is the same. Figure 9-9 shows the KitProg2_Bootloader.cydwr System settings.

Figure 9-9.	KitProg2	Bootloader	System	Settings
i igule 3-3.	TAU TOYZ	Doolloadel	Oystem	oeunga

Option	Value
- Configuration	
- Device Configuration Mode	Compressed
- Enable Error Correcting Code (ECC)	
- Instruction Cache Enabled	
- Enable Fast IMO During Startup	
- Unused Bonded IO	Allow but warn
- Heap Size (bytes)	0x80
- Stack Size (bytes)	0x0800
Include CMSIS Core Peripheral Library Files	
Programming\Debugging	
- Debug Select	GPIO
- Enable Device Protection	
- Embedded Trace (ETM)	
Use Optional XRES	
Operating Conditions	
- Variable VDDA	
- VDDA (V)	5.0
- VDDD (V)	5.0
- VDDIO0 (V)	5.0
- VDDIO1 (V)	5.0
- VDDIO2 (V)	5.0
- VDDIO3 (V)	5.0
Temperature Range	-40C - 85/125C

- 11. Build the project in PSoC Creator by choosing Build > Build Project or pressing [Shift] [F6].
- 12. To program the project onto the PSoC 5LP device, open the Bootloader Host tool, which is available in PSoC Creator. Choose Tools > Bootloader Host, as shown in Figure 9-10. Alternatively, the Bootloader Host tool can be accessed by going to Start > All Programs > Cypress > PSoC Creator 3.3 > Bootloader Host, as shown in Figure 9-11.

Figure 9-10. Open Bootloader Host Tool in PSoC Creator





Figure 9-11. Open Bootloader Host Tool in All Programs



- 13. Press and hold the mode switch while connecting the kit to the computer. If the switch is pressed for more than 100 ms, the PSoC 5LP enters the bootloader. The amber status LED will start blinking to indicate that PSoC 5LP entered bootloader mode.
- 14. In the Bootloader Host tool, click Filters and add a filter to identify the USB device. Ensure that the check box for Show USB Devices is enabled. Set VID as 04B4, PID as F146, and click OK, as shown in Figure 9-12.

Figure 9-12. Port Filters Tab in Bootloader Host Tool

Ports: COM3 - UART Port Filters Active application: No change COM3 - UART Port Filters Active application: No change Community key Old	Bootloader Host File Actions Help	-	
✓ Show BLE Dongles ✓ Show I2C Ports ✓ Show SPI Ports ✓ Show UART Ports ✓ Show USB Devices VID: 0x 04B4 PID: 0x F146 OK Cancel	File: Enter path to cyacd file Ports:	Filters Activ	re application: No change
Generic Serial Port	08:55:16 PM - Selected device: 0	Show BLE Dongles Show I2C Ports Show SPI Ports Show UART Ports Show USB Devices VID: 0x 0484 PID: 0x F146 OK Cancel	9600 • Bits: 8 • Bits: 1 • None •



15. In the Bootloader Host tool, click the **Open File** button (Figure 9-13) to browse to the location of the bootloadable file (*.cyacd), as shown in Figure 9-14. This file is present in the project directory. The .cyacd file is available in the following path:

<Project Directory>\<Project Name.cydsn>\CortexM3\<Compiler Name and Version>\<Debug> or <Release>\<Project Name> 2.cyacd

Note that there are 2 cyacd files in the project directory. *<Project_Name>_1.cyacd* is the first application (in this case the KitProg2 application) and *<Project_Name>_2.cyacd* is the second application (in this case the custom application). Therefore, we want to select *<Project_Name>_2.cyacd* to bootload the custom application.

The "Active application" selection shown in Figure 9-13 is used to determine which application will start once bootloading has completed. In this case, since we have selected Image 2, the custom application will run once bootloading finishes.



Figure 9-13. Open Bootloadable File in Bootloader Host Tool

Figure 9-14. Select <Project_Name_2> .cyacd File from Bootloader Host Tool





- 16. Select the USB Human Interface Device (04B4_F146) USB in the Ports list and click the Program button (Figure 9-13) in the Bootloader Host tool to program the device.
- 17. If the bootload is successful, the log displays "Programming Finished Successfully"; otherwise, it displays "Failed" and a reason for the failure.

Notes:

- The PSoC 5LP pins are connected to the PSoC 5LP GPIO header. These pins are selected to support high-performance analog and digital projects. See section A.1 Pin Assignments on page 78 for pin information.
- Take care when allocating the PSoC 5LP pins for custom applications. For example, P3[2]–P3[3] are dedicated for programming the PSoC 4000S in CY8CKIT-041-40XX. Refer to the respective kit schematics before allocating the pins.
- When a custom bootloadable project is the active application on the PSoC 5LP, the initial capability of the PSoC 5LP to act as a programmer, USB-UART Bridge or USB-I2C Bridge or USB-SPI Bridge is not available. To recover this functionality, switch the active application back to the KitProg2 image.
- The status LEDs do not function unless they are implemented and used by the custom project.

For additional information on bootloaders, refer to the AN73503 – USB HID Bootloader for PSoC 3 and PSoC 5LP.

9.2 Building a Normal Project for PSoC 5LP

A normal project is a new project created for the PSoC 5LP device on the PSoC 4 S-Series Pioneer board. Here, the entire flash of the PSoC 5LP is programmed, overwriting all bootloader and programming code. To recover the programmer, USB-UART Bridge or USB-I2C or USB-SPI Bridge functionality, reprogram the PSoC 5LP device with the factory-set *KitProg2.hex* file, which is shipped with the kit installer.

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. In addition, the PSoC 5LP programming header (PSoC 5LP PROG) on the kit needs to be populated (refer to the kit's Bill of Materials to order the part), to connect the MiniProg3 for programming and debugging the PSoC 5LP device.



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

1. In PSoC Creator, choose File > New > Project and select Target device; select <Launch Device Selector...> from the drop-down list and click Next. See Figure 9-15.

Figure 9-15.	Create New Project in PSoC Creator

Create Project - CY8C5888AXQ-LP096		? ×
Select project type Choose the type of project - design,	library, or workspace.	
Design project: Target kit: Target module: Target device: PSoC 5LP Uibrary project Workspace	 ✓ Last used: CY8C5888AXQ-LP096 Last used: CY8C5888AXQ-LP096 CY8C52LP CY8C54LP CY8C56LP CY8C58LP <launch device="" selector=""></launch> 	
		Next > Cancel

2. Select CY8C5868LTI-LP039, as shown in Figure 9-16. Click OK.

Figure 9-16.	Select the Device
1 19010 0 10.	

🖻 View Datasheet 🖽 Hide/Show Columns 🏷 Reset to Defaults								
		Iview Datasheet ☐ Hide/Show Columns > Reset to Defaults						
ADC CAN 2.0b ADC ADC DAC DAC 12bit DAC Sample and Hold Opamp Analog Blocks Analog Ananol Analog Blocks Analog Analog Blocks Analog Blocks Analog Blocks Analog Blocks Analog Blocks Analog Blocks Analog Blocks Analog Ananol Analog Blocks Analog Analog Blocks Analog Analog Anal	Multi-Function Timers (MFT)	Multi-Function Serial (MFS)	*					
Filters:								
CY8C5868AXI-LP032 - 2x 12-bit SAR 1x 20-bit Delta Sigma 4 - - 4 4x SC/CT - 4 1 24 - 1 0.	5 -	-						
CY8C5868AXI-LP035 1 2x 12-bit SAR 1x 20-bit Delta Sigma 4 - - 4 4x SC/CT - 4 1 24 - 1 0.	5 -	-						
CY8C5868LTI-LP036 - 2x 12-bit SAR 1x 20-bit Delta Sigma 4 - - 4 4x SC/CT - 4 1 24 - 1 0.	5 -	-						
CY8C5868LTI-LP038 - 2x 12-bit SAR 1x 20-bit Delta Sigma 4 - - 4 4x SC/CT - 4 1 24 - 1 0.	5 -	-	E					
CY8C5868LTI-LP039 1 2x 12-bit SAR 1x 20-bit Delta Sigma 4 - - 4 4x SC/CT - 4 1 24 - 1 0.0	5 -							
CY8C5888AXI-LP096 1 2x 12-bit SAR 1x 20-bit Delta Sigma 4 - - 4 4x SC/CT - 4 1 24 - 1 0.	5 -	-	Ŧ					
66 of 66 devices found Clear Filters								
ОК	C	ancel						

- 3. Develop your custom project.
- 4. Build the project in PSoC Creator by choosing Build > Build Project or pressing [Shift] [F6].
- 5. Connect the MiniProg3 to the onboard PSoC 5LP Programming header J11 (which needs to be populated).



 To program the PSoC 5LP with PSoC Creator, choose Debug > Program or press [Ctrl] [F5]. If the Select Debug Target window appears and shows MiniProg3 and the selected device in the project under it (CY8C5868LTI-LP039), select the device and click Connect to acquire and program.

If the PSoC device is not available, click the **Port Setting** button. Set **Active Protocol** to 'SWD', **Acquire Mode** to 'Power Cycle', and **Connector** to either 5 pin or 10 pin depending on the kit. Then, click the **Port Acquire** button for the PSoC 5LP device to appear. See Figure 9-17.



Select Debug Target	-7	Select Debug Target		7 💌
MiniProg3/1427DD0005F0	MiniProg3/1427DD0005F0	E-5 MiniProg3/14		PSoC 5LP CY8C5868LTI-LP039
	POWER = 3 VOLTAGE_ADC = 1880 FREQUENCY = 1600000 CONNECTOR = 10 PROTOCOL = SWD	PSoC SLP (CY8C5868LTI-LP039 (Connected)	PSoC 5LP (ARM CM3) Silicon ID: 0x28A01477 Cypress ID: 0x2E127069 Revision: PRODUCTION
Show all targets	MiniProg3 version 2.05 [3.08/2.08] Port Setting Port &cquire	Show all targets		Target acquired
	ОК			ок
	OK			UN
	Active Pateols.	SWD •		
	[OK Canoel		

Notes:

- The PSoC 5LP programming header is not populated.
- The PSoC 5LP pins are brought to the PSoC 5LP GPIO header. These pins are selected to support high-performance analog and digital projects. See A.1 Pin Assignments on page 78 for pin information.
- Take care when allocating the PSoC 5LP pins for custom applications. For example, P3[2]–P3[3] are dedicated for programming the PSoC 4000S in CY8CKIT-041-40XX. Refer to the respective kit schematics before allocating the pins.
- 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 or USB-SPI Bridge is not available.
- The status LEDs do not function unless they are implemented and used by the custom project.

10. Troubleshooting the KitProg2



This section explains the methods to troubleshoot the KitProg2 and recover the KitProg2 firmware if you modified it.

10.1 KitProg2 Status LED Indication

The KitProg2 status LEDs on the development kit indicate the status of the KitProg2 operation using different blink rates. Table 10-1 shows the KitProg2 LED indication and the corresponding status of the KitProg2.

User Indication		Scenario	Action Dequired by Llear	
Amber LED	Green LED	Red LED	Scenario	Action Required by User
ON	OFF	OFF	USB enumeration is successful. KitProg2 is in PPCOM programming mode.	PSoC Creator, PSoC Programmer, BCP, and any serial port terminal program can use the kit functions. Normal operation of your application on the target device occurs in this mode.
OFF	OFF	OFF	USB enumeration is unsuccessful.	This indicates that the USB enumeration was unsuccessful. It may happen if the kit is not powered from the USB host. Verify the USB cable and check if PSoC Programmer is installed on the PC.
ON	Blinking Frequency = 8 Hz	OFF	Programming of the target device is currently in process.	No action is required.
ON	ON	OFF	Programming of target successful	No action is required.
ON	OFF	ON	Programming of target not successful	Use correct hex file with respect to the device on board
OFF	OFF	OFF	KitProg2 is in mass storage programming mode/CMSIS-DAP programming mode	Open KitProg2 drive. Drag and drop a hex file that corresponds to the target device to the drive window to start programming. Normal operation of your application on the target device occurs in this mode.

Table 10-1. Meaning of KitProg2 LED Indications



User Indication		Scenario	Action Required by User	
Amber LED	Green LED	Red LED	Scenario	Action Required by User
OFF	8 Hz	OFF	Programming of the target device is currently in process in Mass storage Mode.	No action is required.
OFF	ON	OFF	Programming of the target device is complete and successful in mass storage mode. Note that this is applicable only in drag-and-drop programming. The green LED will not turn ON if the programming is through PSoC Creator or PSoC Programmer.	No action is required. Normal operation of your application on the target device occurs in this mode.
Blinking Frequency = 1 Hz	OFF	OFF	KitProg2 is in bootloader mode.	In this mode, you can bootload a new version of your custom application firmware.
8 Hz (Blinking)	User defined	User Defined	Custom application mode	No Action required
ON	Blinking Frequency = 15.0 Hz	OFF	SWD or I2C operation is in progress. The kit's COM port connect and disconnect event (only one blink).	In PSoC Programmer, watch the log window for status messages for SWD operations. In the BCP, the LED blinks on I2C command requests. In BCP or any other serial port terminal program, the kit's COM port will appear when the port is connected and it will disappear when the port is disconnected.

Table 10-1.	Meaning of KitProg2 LED Indications ((continued)

Note: There is another category of Kits known as prototyping Kits which have single button and a single LED based KitProg2 System. To enter into various modes below points should be noted:

- 1. When the kit is in KitProg2 mode, the Amber LED will be on.
- When the kit is in Mass Storage/CMSIS DAP mode, the Amber LED is turned off. To reach this mode, press and release the mode switch provided on the prototyping kit in less than 2 seconds.
 Note: In 1.04 KitProg2 version, Amber LED shows breathing effect in Mass Storage or CMSIS DAP Mode



3. When the kit is in Bootloader mode, the Amber LED shows a blinking effect. To reach this mode, hold down the mode switch and insert the kit into a USB port.

Note: The various modes and their representation in terms of LEDs with different colors are not supported in these categories of kits eg. CY8CIT-146 PSoC 4200DS Prototyping Kit.

Note: The BCP software cannot connect to the KitProg2, if the KitProg2 firmware version is outdated. See Updating the KitProg2 Firmware on page 17 to update the KitProg2 firmware.

Note: The programming/debugging function and USB-I2C Bridge/USB-SPI Bridge function of the KitProg2 are mutually exclusive functions and cannot be used together. Therefore, to use one function the other function should be disconnected. For instance, to program the device while using the USB-I2C Bridge/USB-SPI Bridge in BCP, either close BCP or disconnect the USB-I2C Bridge/USB-SPI Bridge. The USB-UART Bridge function of the KitProg2, however, can run in parallel to both programming/debugging and USB-I2C Bridge/USB-SPI Bridge functions. USB-I2C Bridge and USB-SPI Bridge functionality cannot be used simultaneously.

10.2 PSoC 5LP Factory Program Restore Instructions

10.2.1 PSoC 5LP is Programmed with a Custom Application

Reprogramming the PSoC 5LP device with a new flash image in the KitProg2 area of flash (i.e. application area 1) will forfeit the ability to use the PSoC 5LP device as a programmer/debugger for the kit. See section 10.2.1.1 Restore PSoC 5LP Factory Program Using PSoC Programmer on page 74, for details on restoring the KitProg2.

Note: This method cannot be used to recover the KitProg2 if the PSoC 5LP was reprogrammed using a MiniProg3. See section 10.2.2 Restore PSoC 5LP KitProg2 using MiniProg3 on page 76 if you want to recover the KitProg2 functionality using a MiniProg3.



10.2.1.1 Restore PSoC 5LP Factory Program Using PSoC Programmer

- 1. Launch PSoC Programmer from Start > Cypress > PSoC Programmer <version> > PSoC Programmer <version>.
- 2. Configure the Kit in bootloader mode. To do this, while pressing the mode switch, connect the Kit to the computer using the included USB cable (USB Standard-A to Micro-B). This puts the PSoC 5LP into bootloader mode, which is indicated by the blinking amber status LED.
- 3. The following message appears in the PSoC Programmer **Results** window (see Figure 10-1): "KitProg2 Bootloader devices are detected".

Figure 10-1. PSoC Programmer Results Window

PSoC Programmer		
File View Options He	łp	
🖆 · 🗼 🔘 BB		
Port Selection	Programmer Utilities JTAG	
	Programming Parameters	
	File Path: C:\Program Files (x86)\Cypre	ess\Programmer\KitProg2.hex
	Programmer:	
	Programming Mode: Reset Power Cycle	Power Detect
	Verification: On Off 	Connector: 5p @ 10p
Device Family	AutoDetection: On Off	Clock Speed: 1.6 MHz V
CY8C5xxLP -	Programmer Characteristics	Status
Device	Protocol: O JTAG O SWD O ISSP O I2C	Execution Time: Power Status:
CY8C5868LTI-LP039	Voltage: 5.0 V 3.3 V 2.5 V 1.8 V	Voltage: NA
Actions	Results	
Connected at 4:29:20	5 PM KitProg2 bootloader devices are	detected
	Please close all ports, then na tab and click the Upgrade Firmw Bridge	
For Help, press F1	·····	Not Connected



4. Switch to the **Utilities** tab in PSoC Programmer and click the **Upgrade Firmware** button, as shown in Figure 10-2. Unplug all other PSoC programmers (such as MiniProg3 and DVKProg) from the PC before clicking the **Upgrade Firmware** button.

Figure 10-2. Upgrade Firmware



5. After programming is completed, the message "Firmware Update Finished at <time>" appears, and PASS message is indicated on the status bar, as shown in Figure 10-3.

PSoC Programmer
File View Options Help
Port Selection IV Programmer Utilities JTAG
KitProg20A2116F7000454(Upgrade Firmware Click to upgrade connected device's firmware
Erase Block Click to erase user specific flash block
Device Family CY8C4Doc-S
Device
CY8C4045AZI-S413 v
Actions Results
KitProg2 Version 1.01 [HW Rev.0x01]
Firmware Update Finished at 1:30:02 PM
at 1:30:02 PM Succeeded
Verifying
Upgrading
Initializing
Firmware Upgrade Started at 1:29:45 FM
at 1:25:45 PM Firmware Upgrade
Requested at 1:29:45 PM
For Help, press F1 Connected

Figure 10-3. Firmware Update Completed

6. The factory program is now successfully restored on the PSoC 5LP. It can be used as the programmer/debugger for the PSoC 4000S device.



10.2.2 Restore PSoC 5LP KitProg2 using MiniProg3

This section explains the method to reprogram the PSoC 5LP using a MiniProg3 to recover the KitProg2 functionality. This method must be used to recover the KitProg2 if the PSoC 5LP was completely reprogrammed. Note that this method of restoring KitProg2 will erase any custom applications loaded earlier.

- 1. Launch PSoC Programmer from Start > Cypress > PSoC Programmer <version> > PSoC Programmer <version>.
- 2. Connect the MiniProg3 to the PC. Connect the 5-pin or 10-pin connector (depending on the kit) of the MiniProg3 to the onboard PSoC 5LP programming header.

Note: This header is not populated by default. You will need to populate it to connect the MiniProg3.

- 3. Select MiniProg3 from the Port Selection list in PSoC Programmer on your PC.
- 4. Using the File > Open menu or the File Load icon, load the KitProg2.hex file, which is installed with the kit software, as shown in Figure 10-4. The default location for this file is: C:\Program Files (x86)\Cypress\Programmer\KitProg2.hex.

PSoC Program									
File View	Options Help								
🕋 · 🍡	o BB 🕻 🗎								
Port Selection	ton 😼 Open HEX file								
MiniProg3/									
	Organize 👻 New fold	ler)iii 🔹 🗍	0					
	☆ Favorites	Name	Date modified T	ype 🔺					
	A HURONES	퉬 Service	1/19/2016 5:03 PM F	ile fol					
0	libraries	퉬 Updater	1/19/2016 5:02 PM F	ile fol					
Device Family CY8C5ioxLP		Updates	1/19/2016 5:02 PM F	ile fol					
	Computer	DVKProg1.hex	1/14/2016 9:53 AM	IEX Fil					
Device		helper.hex	1/14/2016 9:52 AM	IEX Fil					
CY8C5868LT	🗣 Network	KitProg.hex	1/14/2016 9:53 AM	IEX Fil					
Actions		KitProg2.hex		IEX Fil					
		minifirmware.hex	1/14/2016 9:52 AM H	IEX Fil					
		TrueTouchBridge.hex	1/18/2016 10:09 AM	IEX Fil					
		usbtoiic.hex	1/14/2016 9:52 AM H	IEX Fil 👻					
		•		•					
	File	ame: KitProg2.hex	✓ HEX files (*.hex)	-					
			Open - Cance						
			Open V Cance	·					
or Help, press F1			FAIL Not Power	Connected					

Figure 10-4. Select the KitProg2.hex File to Program the PSoC 5LP

- 5. Select the **Power Cycle** option for Programming Mode, **5.0 V** for voltage, **10p** (or 5p, if applicable) for Connector, and **SWD** for Protocol.
- 6. Click the **Program** button or **File > Program** to program the PSoC 5LP device.



7. After programming is complete, the "Program Finished at <time>" message is displayed, and PASS is indicated on the status bar, as shown in Figure 10-5.

Figure 10-5. Firmware Programming Completed

PSoC Programmer		- • ×		
File View Options Help				
🖆 🔰 🗿 🔒				
Port Selection	Programmer Utilities JTAG			
MiniProg3/1427DD0005F0	Programming Parameters File Path: C:\Program Files (x86)\Cypress\Programmer\KitProg2.hex Programmer: MiniProg3/1427DD0005F0			
	Programming Mode: Reset Power Cycle Power Detect			
	Verification: ● On ● Off Connector: ● 5p ● 10p AutoDetection: ● On ● Off Clock Sneed: 16 MHz			
Device Family				
CY8C5xxLP v	Programmer Characteristics Protocol: ITAG SWD ISSP 120 Execution Time: 18.8 seconds			
Device CY8C5868LTI-LP039 ~	Protocol: ◯ JTAG ● SWD ◯ ISSP ◯ I2C Execution Time: To.s seconds Voltage: ● 5.0 V ◯ 3.3 V ◯ 2.5 V ◯ 1.8 V Voltage: 220 mV			
Actions	Results	^		
Program Finished at 6:13:44 PM Programming Succeeded Doing Checksum Doing Protect Programming of Flash Succeeded Programming of Flash Starting Erase Succeeded				
Device set to CY8C5868LTI-LP039 at 262144 FLASH bytes 6:13:37 PM Device Family set to				
or Help, press F1	PASS Not Powered	Connected		

A. Appendix



A.1 Pin Assignments

A.1.1 PSoC 5LP GPIO and Custom Application Header (J8 and J11) for CY8CKIT-041-40XX and CY8CKIT-041-41XX

J8								
Pin	PSoC 5LP Signal	PSoC 5LP Description	Pin	PSoC 5LP Signal	PSoC 5LP Description			
J8_01	PSoC 5LP_VDD	VDD	J8_02	N.C	No Connect			
J8_03	P15[1]	SPI MOSI	J8_04	P0[1]	CUSTOM			
J8_05	P15[2]	SPI SCLK	J8_06	P12[0]	I2C SCL			
J8_07	P15[3]	SPI SSEL	J8_08	P3[0]	CUSTOM			
J8_09	P12[5]	SPI MISO	J8_10	P12[1]	I2C SDA			
J8_11	P3[4]	CUSTOM	J8_12	P3[5]	CUSTOM			
J8_13	P12[7]	UART RX	J8_14	P12[6]	UART TX			
J8_15	P3[6]	CUSTOM	J8_16	GND	GND			
	J11							
Pin	PSoC 5LP Signal	PSoC 5LP Description	Pin	PSoC 5LP Signal	PSoC 5LP Description			
J11_01	P0[2]	CUSTOM	J11_02	PSoC 5LP_VBUS	VBUS			
J11_03	P0[3]	CUSTOM	J11_04	GND	GND			
J11_05	P0[7]	CUSTOM	J11_06	PSoC 5LP_XRES	XRES			
J11_07	P1[6]	INT_PIN	J11_08	PSoC 5LP_SWDCLK	SWD CLK			
J11_09	P15[4]	SUSPEND	J11_10	PSoC 5LP_SWDIO	SWD IO			



A.1.2 PSoC 5LP GPIO and Custom Application Header (J16 and J11) for CY8CKIT-048

J16								
Pin	PSoC 5LP Signal	PSoC 5LP Description	Pin	PSoC 5LP Signal	PSoC 5LP Description			
J16_01	PSoC 5LP_VDD	VDD	J16_02	N.C	No Connect			
J16_03	P15[1]	SPI MOSI	J16_04	P0[1]	CUSTOM			
J16_05	P15[2]	SPI SCLK	J16_06	P12[0]	I2C SCL			
J16_07	P15[3]	SPI SSEL	J16_08	P3[0]	CUSTOM			
J16_09	P12[5]	SPI MISO	J16_10	P12[1]	I2C SDA			
J16_11	P3[4]	CUSTOM	J16_12	P3[5]	CUSTOM			
J16_13	P12[7]	UART RX	J16_14	P12[6]	UART TX			
J16_15	P3[6]	CUSTOM	J16_16	GND	GND			
	J11							
Pin	PSoC 5LP Signal	PSoC 5LP Description	Pin	PSoC 5LP Signal	PSoC 5LP Description			
J11_01	P0[2]	CUSTOM	J11_02	PSoC 5LP_VBUS	VBUS			
J11_03	P0[3]	CUSTOM	J11_04	GND	GND			
J11_05	P0[7]	CUSTOM	J11_06	PSoC 5LP_XRES	RST			
J11_07	P1[6]	INT_PIN	J11_08	PSoC 5LP_SWDCLK	SWD CLK			
J11_09	P15[4]	SUSPEND	J11_10	PSoC 5LP_SWDIO	SWD IO			

A.1.3 PSoC 5LP GPIO and Custom Application Header (J6 and J7) for CY8CKIT-145-40XX

	J6								
Pin	PSoC 5LP Signal	PSoC 5LP Description	Pin	PSoC 5LP Signal	PSoC 5LP Description				
J6_01	PSoC 5LP_VBUS	VBUS	J6_02	GND	GND				
J6_03	P12[5]	CUSTOM	J6_04	P12[0]	I2C SCL				
J6_05	P12[1]	I2C SDA	J6_06	P12[7]	UART_RX				
J6_07	P12[6]	UART_TX							
			J7						
Pin	PSoC 5LP Signal	PSoC 5LP Description	Pin	PSoC 5LP Signal	PSoC 5LP Description				
J7_01	GND	GND	J7_02	P3[0]	CUSTOM				
J7_03	P3[4]	CUSTOM	J7_04	P3[5]	CUSTOM				
J7_05	P3[6]	CUSTOM	J7_06	P0[2]	CUSTOM				
J7_07	P0[1]	CUSTOM							

Note: For other kits, refer to the kit user guide to get this information.

Revision History



Document Revision History

	Document Title: KitProg2 User Guide					
Document Number: 002-10738						
Revision	ECN Number	Issue Date	Origin of Change	Description of Change		
**	5097455	01/22/2016	VJYA/ SNVN	Initial version of KitProg2 User Guide.		
*A	5201134	04/01/2016	VJYA	Updated Introduction chapter on page 5:		
				Added "Acronyms" on page 9.		
				Updated Appendix chapter on page 78:		
				Updated "Pin Assignments" on page 78:		
				Added "PSoC 5LP GPIO and Custom Application Header (J16 and J11) for CY8CKIT-048".		
				Added "PSoC 5LP GPIO and Custom Application Header (J6 and J7) for CY8CKIT-145-40XX".		
*В	5268884	05/12/2016	VJYA	Added "CY8CKIT-048 PSoC Analog Coprocessor Pioneer Kit" related information in all instances across the document.		
				Added "CY8CKIT-145-40XX PSoC 4 S-Series Prototyping Kit" related information in all instances across the document.		
*C	5392257	08/05/2016	SRDS	Added "CY8CKIT-041-41XX PSoC 4100S Pioneer Kit" related information in all instances across the document.		
*D	5660193	03/24/2017	NMIT / VKVK	Added "USB-SPI Bridge" related information in all instances across the document.		
				Updated Introduction chapter on page 5:		
				Updated description.		
				Updated "Switching between KitProg2 Modes" on page 7:		
				Updated description.		
				Updated Figure 1-3.		
				Updated Ecosystem chapter on page 10:		
				Updated Table 2-1.		
				Updated Table 2-3.		



Document Revision History (continued)

Document Number: 002-10738

Revision	ECN Number	Issue Date	Origin of Change	Description of Change
*D (cont.)	5660193	03/24/2017	NMIT /	Updated KitProg2 Mode Programmer and Debugger chapter on page 12:
			VKVK	Updated description.
				Updated "Programming Using PSoC Creator" on page 13:
				Updated description.
				Updated Figure 3-4.
				Updated Figure 3-5.
				Updated "Debugging Using PSoC Creator" on page 16:
				Updated description.
				Updated Figure 3-8.
				Removed "Programming and Debugging using µVision".
				Added CMSIS-DAP Mode Programming and Debugging chapter on page 20.
				Updated Mass Storage Programmer chapter on page 27:
				Updated description.
				Updated "Enter or Exit the Mass Storage Programmer Mode" on page 27:
				Updated description.
				Updated "Programming Using the Mass Storage Programmer" on page 27: Updated description.
				Updated "Frequently Asked Questions on KitProg2 Mass Storage Programmer" on page 29:
	Updated description. Updated USB-UART Bridge chapter on p	Updated description.		
				Updated USB-UA
				Updated description.
			Updated Figure 6-1	
				Updated USB-I2C Bridge chapter on page 41:
				Updated description.
				Updated Figure 7-1.
				Updated Figure 7-11.
				Added USB-SPI Bridge chapter on page 52.



Document Revision History (continued)

Document Number: 002-10738

Revision	ECN Number	Issue Date	Origin of Change	Description of Change
*D (cont.)	5660193	0193 03/24/2017 NMIT / VKVK	Updated Developing Applications for PSoC 5LP chapter on page 58:	
			VKVK	Updated description.
				Updated "Building a Custom Bootloadable Application for KitProg2" on page 58:
				Updated description.
				Updated Figure 9-1.
				Updated Figure 9-2.
				Updated Figure 9-6.
				Updated Figure 9-10.
				Updated "Building a Normal Project for PSoC 5LP" on page 67:
				Updated description.
				Updated Figure 9-15.
				Updated Figure 9-16.
				Updated Troubleshooting the KitProg2 chapter on page 71:
			Updated "KitProg2 Status LED Indication" on page 71:	
				Updated Table 10-1.
				Updated description.
				Updated "PSoC 5LP Factory Program Restore Instructions" on page 73:
				Updated "PSoC 5LP is Programmed with a Custom Application" on page 73:
				Updated description.
				Updated "Restore PSoC 5LP KitProg2 using MiniProg3" on page 76:
				Updated description.
				Updated Appendix chapter on page 78:
				Updated "Pin Assignments" on page 78:
				Updated description.
				Updated to new template.
*E	5767710	06/08/2017	NMIT	Updated Troubleshooting the KitProg2 chapter on page 71:
				Updated "KitProg2 Status LED Indication" on page 71:
				Updated Table 10-1.
*F	5940717	10/23/2017	NMIT	Updated Ecosystem chapter on page 10:
				Updated Table 2-1 (Added new kit CY8CKIT-149 to support S3 device).
				Updated Troubleshooting the KitProg2 chapter on page 71:
				Updated "KitProg2 Status LED Indication" on page 71:
				Updated Table 10-1.
*G	5960991	11/08/2017	NMIT	Fixed typos (Replaced "CMIS" with "CMSIS" and replaced "CMISIS" with "CMSIS" in all instances across the document).



Document Revision History (continued)

	Document Title: KitProg2 User Guide Document Number: 002-10738						
Revision	ECN Number	Issue Date	Origin of Change	Description of Change			
*G (cont.)	5960991	11/08/2017	NMIT	Updated Introduction chapter on page 5:			
				Updated "Switching between KitProg2 Modes" on page 7:			
				Updated description.			
				Updated KitProg2 Mode Programmer and Debugger chapter on page 12: Updated description.			
				Updated CMSIS-DAP Mode Programming and Debugging chapter on page 20:			
				Updated description.			
				Updated "Programming and Debugging using µVision" on page 20:			
				Updated description.			
				Updated Mass Storage Programmer chapter on page 27:			
				Updated "Enter or Exit the Mass Storage Programmer Mode" on page 27:			
				Updated description.			
				Updated Troubleshooting the KitProg2 chapter on page 71:			
				Updated "KitProg2 Status LED Indication" on page 71:			
				Updated Table 10-1.			
				Updated description.			
*H	6022878	01/10/2018	NMIT	Updated Troubleshooting the KitProg2 chapter on page 71:			
				Updated "KitProg2 Status LED Indication" on page 71:			
				Updated Table 10-1.			
				Completing Sunset Review.			
*I	6043479	03/07/2018	NMIT	Updated Ecosystem chapter on page 10:			
				Updated Table 2-1.			
				Updated Developing Applications for PSoC 5LP chapter on page 58:			
				Updated "Building a Custom Bootloadable Application for KitProg2" on page 58:			
				Updated description.			
				Added Figure 9-8.			
				Updated Figure 9-9.			