mail

Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from, Europe, America and south Asia, supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of "Quality Parts, Customers Priority, Honest Operation, and Considerate Service", our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip, ALPS, ROHM, Xilinx, Pulse, ON, Everlight and Freescale. Main products comprise IC, Modules, Potentiometer, IC Socket, Relay, Connector. Our parts cover such applications as commercial, industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!



Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832 Email & Skype: info@chipsmall.com Web: www.chipsmall.com Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China





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 | 9 |
| 2. | Ecosys | tem | 10 |
| 3. | KitProg | 2 Mode Programmer and Debugger | 12 |
| | 3.1 | KitProg2 Driver Installation | 12 |
| | 3.2 | Programming Using PSoC Creator | 13 |
| | 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 | 27 |
| | 5.3 | Frequently Asked Questions on KitProg2 Mass Storage Programmer | 29 |
| 6. | USB-U/ | ART Bridge | 30 |
| 7. | USB-I2 | C Bridge | 41 |
| 8. | USB-SF | PIBridge | 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 | 67 |
| 10 |). Troub | leshooting the KitProg2 | 71 |
| | 10.1 | KitProg2 Status LED Indication | 71 |
| | 10.2 | PSoC 5LP Factory Program Restore Instructions | 73 |
| | | 10.2.1 PSoC 5LP is Programmed with a Custom Application | 73 |
| | | 10.2.1.1 Restore PSoC 5LP Factory Program | |
| | | Using PSoC Programmer | 74 |
| | | 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 |
| 12C | 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

| 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 |



| 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)



- 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

| I | | Build CY8CKIT_041_Proximity_Sensing Shift+F6 | |
|----|-----------|---|--|
| | | Clean CY8CKIT_041_Proximity_Sensing | |
|) | ** | Clean and Build CY8CKIT_041_Proximity_Sensing | |
| | ž | Cancel Build Ctrl+Break | |
| P | ٨ | Compile <u>File</u> Ctrl+F6 | |
| 1 | 1 | Generate Application | |
| ir | | 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 | 8 × |
|-------------------------------|--|
| = 🕤 KitProg2/1D1A18EB02105400 | KitProg2/1D1A18EB02105400 |
| PSoC 4000S CY8C4045AZI-S413 | POWER = 3 VOLTAGE_ADC = 4812 FREQUENCY = 2000000 PROTOCOL = SWD |
| | KtProg2 Version 0.07 [HW Rev.0x01] |
| Show all targets | Port Setting Port Acquire |

 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

| Debug | | Tools | $\underline{W} indow$ | <u>H</u> elp | | |
|-------|----------------------------------|------------|-----------------------|--------------|-----------|---|
| | Windows | | | | | ۲ |
| | Pro | gram | | | Ctrl+F5 | |
| ž | Sel | ect Debu | g <u>T</u> arget | | | |
| 羡 | Debug F5 | | | | | |
| 燕 | Debug without Programming Alt+F5 | | | | | |
| | Attach to Running Target | | | | | |
| ø | To | ggle Brea | kpoint | | F9 | |
| | New Breakpoint | | | • | | |
| 2 | Del | ete All Bi | rea <u>k</u> points | Ctrl | +Shift+F9 | |
| 0 | Enable All Breakpoints | | | | | |

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



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.

- 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 | ogrammer Utilities JTAG |
| KitProg2/0B0C05B3031054 | Upgrade Firmware Click to upgrade connected device's firmware |
| 1 | Erase Block Click to erase user specific flash block |
| Device Family | |
| CY8C40x-S * | |
| Device | |
| CY8C4045AZI-S413 * | |
| Actions | Results |
| Port Opened with Warnings at 3:39:31 PM Opening Port at 3:39:2 PM | Please navigate to the Utilities tab and click the Upgrade Firmware button KitProg2 version Expecting 1.01, but found 1.00. |
| Connected at 3:39:28 F | M KitProg2/0B0C05B303105400 |
| | |
| or Help, press F1 | FAIL Powered 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

| File View Options Help | | |
|--|---------------------------|---------------------------------------|
| 🕋 · 🗼 🔘 BB 🚽 | 9 🖹 🗅 🖹 🔘 |) |
| Port Selection | ogrammer Utilities JTAG | |
| KitProg2/0A2116F7000454(| Upgrade Firmware Click to | o upgrade connected device's firmware |
| | Erase Block Click to | o erase user specific flash block |
| Device Family | | |
| CY8C40xx-S * | | |
| Device | | |
| CY8C4045AZI-S413 - | | KitProg2 firmware version |
| Actions | Results | |
| | KitProg2 Version | 1.01 [HW Rev.0x01] |
| Firmware Update Finish at 1:30:02 PM | ed Succeeded | KitProg2 firmware upgrade message |
| | Verifying | |
| | Upgrading | |
| Firmurre Hograde Start | Initializing | |
| at 1:29:45 PM | cu | |
| firmware Upgrade Requested at 1:29:45 P | М | |
| NIP. | | |

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

| 💁 🔁 | | |
|-----------|---------------------------------|---------------|
| 🔯 Workspa | ace 'CapSense_LowPower' (1 Proj | jects) |
| E-Pa Proi | Set As Active Project | BC 4045 A71-S |
| H | Add | • |
| | Build CapSense_LowPower | |
| e-1 | Clean CapSense_LowPower | |
| 🕮 | Clean and Build CapSense_Low | Power |
| | Update Components | |
| Eb | <u>C</u> opy | Ctrl+C |
| 63 | Paste | Ctrl+V |
| | Save CapSense_LowPower As | |
| | Remove From Workspace | |
| | Rename | F2 |
| | Unload Project | |
| | Dependencies | |
| | Build Order | |
| | Device Selector | |
| | Archive Workspace/Project | |
| | 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

| Select Toolchain Choose the desired to | ol chain to build with in μ Vision | |
|---|--|----|
| Select target toolchain: | ARM MDK Generic ARM MDK Generic ARM GCC Generic | |
| | | |
| r | | 11 |

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

| Select the files | |
|-----------------------------|--|
| Select All | Unselect All |
| Name | Path |
| v main.c v cyapicallback | s.h G:\Projects\Kit_Prog\Projects\Design01\CapSense_LowPower.cydsn\r s.h G:\Projects\Kit_Prog\Projects\Design01\CapSense_LowPower.cydsn\r |
| | |

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 | |
| New Op Sav Clo | v µVision v Multi-l en Project e Project se Project | n Project Project Wo ct : in µVisio ct | orkspace n4 format | | | | | |
| Exp Ma | ort nage | | | | | | | • |
| Sel | e ct Devic nove Iter | e for Targ | et 'CapSense_ | LowPow | /er' | | | |
| 💦 Ор | tions for | Target 'C | apSense_LowF | ower' | | | | Alt+F7 |
| Cla | an targe | + <i>r</i> | | | | | | |

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: ○ | 🗆 RAM1: |
| □ ROM2: □ C | □ RAM2: □ |
| E ROM3: | RAM3: |
| on-chip | on-chip |
| IROM1: 0x0 0x4000 € | ✓ IRAM1: 0x20000000 0x800 □ |
| IROM2: C | IRAM2: |
| | |
| OK Can | 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.

| · Use Targe | t Driver for Flash Programming | ☐ Use Debug Driver |
|---|--|---|
| | CMSIS-DAP Debugger | Settings Update Target before Debugging |
| Init File: C Use Extern Command: Arouments | ULINK2/ME Cortex Debugger Stellaris ICDI Signum Systems JTAGjet J-LINK / J-TRACE Cortex ULINK Pro Cortex Debugger NULINK Debugger ST-Link Debugger ST-Link Debugger | Edt] |
| | CMSIS-DAF Debugger Fast Models Debugger Cypress MiniProg3/KitProg v3.3 | |
| Configure Imag | e File Processing (FCARM): | |
| Output File: | | Add Output File to Group: |
| | | Generated_Header |
| Image Files R | not Folder | Generate Listing |

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

| -M Target Driver Setup | an inserie | an interim | and the second s | U |
|--|---|-----------------------------|--|---|
| Download Function Composed Function C Erase Full Chip C Erase Sectors C Do not Erase | Program Profram Verify Reset and Ri | RAM for Al Start: Ox | gorithm 20000200 Size: 0x0600 | |
| Programming Algorithm Description | Device Size | Device Type | Address Range | |
| CY8C40xx (16kB) Flash | 16k | On-chip Flash | 00000000H - 00003FFFH | |
| CY8C40xx (16kB) Flash | 16k | On-chip Flash Start: | 00000000H - 00003FFFH | |
| CY8C40xx (16kB) Flash | 16k Add | On-chip Flash Start: Remove | 00000000H - 00003FFFH Size: | |



| | | | Edit | |
|--|---|---|--|-------|
| x-M Target Driver Setup | 100 million (100 million) | | | |
| Dug Flash Download | | | | |
| Download Function | | BAMfor | Noorthm | |
| Enon C Erase Full Chip | Program | | *gont in | |
| Erase Sectors | Verify | Start: | x20000200 Size: 0x060 | 00 |
| C Do not Erase | Reset and F | Run | | |
| | | | | |
| Programming Algorithm | | | | |
| Description | Device Size | Device Type | Address Range | |
| Add Elect Deservoire Ale | a data a | The street Prints | Statement Statement | |
| Add Flash Programming Aig | onunm | | | |
| | | <u></u> | | |
| | | | | |
| Description | Flash Size | Device Type | Origin | ^ |
| Description ATSAMR21 128kB Flash | Flash Size 128k | Device Type On-chip Flash | Origin MDK Core | ^ |
| Description ATSAMR21 128kB Flash ATSAMR21 256kB Flash | Flash Size 128k 256k | Device Type On-chip Flash On-chip Flash | Origin MDK Core MDK Core | |
| Description ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 64kB Flash | Flash Size 128k 256k 64k | Device Type On-chip Flash On-chip Flash On-chip Flash | Origin MDK Core MDK Core MDK Core | ^ |
| Description ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 64kB Flash CC2538xx 128 KB | Flash Size 128k 256k 64k 128k | Device Type On-chip Flash On-chip Flash On-chip Flash On-chip Flash | Origin MDK Core MDK Core MDK Core MDK Core | ^ |
| Description ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 64kB Flash CC2538x 128 KB CC2538x 256 KB | Flash Size 128k 256k 64k 128k 256k | Device Type On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash | Origin MDK Core MDK Core MDK Core MDK Core MDK Core | ^ |
| Description ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 64kB Flash CC2538xx 128 KB CC2538xx 256 KB CC2538xx 256 KB | Flash Size 128k 256k 64k 128k 256k 512k | Device Type On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash | Origin MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core | ^ |
| Description ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 64kB Flash CC2538x 128 KB CC2538x 512 KB CC2538x 512 KB CC9528x 512 KB | Flash Size 128k 256k 64k 128k 256k 512k 16k | Device Type On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash | Origin MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core | |
| Description ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 64kB Flash CC2538x 256 KB CC2538x 256 KB CC2538x 251 kB GY804Dxx (16kB) Flash CY804Dxx (16kB) Flash | Flash Size 128k 256k 64k 128k 256k 512k 16k | Device Type On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash | Origin MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core MDK Core | ^ |
| Description ATSAMR21 128kB Flash ATSAMR21 258kB Flash ATSAMR21 64kB Flash CC2538x 128 KB CC2538x 256 KB CC2538x 256 KB CC2538x 256 KB CC2538x 256 KB CC2538x 256 KB CC2538x 250 KB CC2538x 250 KB CC25500 Flash | Flash Size 128k 256k 64k 128k 256k 512k 16k 32k 256k | Device Type On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash | Origin MDK Core | ^ |
| Description ATSAMR21 128kB Flash ATSAMR21 256kB Flash ATSAMR21 64kB Flash CC2538x 128 KB CC2538x 512 KB CC2538xc 512 KB CC2538xc 512 KB CC2538xc 512 KB CC253xc 718 kB CC250x Flash CY8C42xc TMACRO (32kB) CY8C50xx Configuration | Flash Size 128k 256k 64k 128k 256k 512k 16k 32k 32k | Device Type On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash On-chip Flash | Origin MDK Core | ^ |
| Description ATSAMR21 128kB Flash ATSAMR21 258kB Flash ATSAMR21 258kB Flash CC2538x 128 kB CC2538x 256 KB CC2538x 256 KB CC2538x 251 kB CY8C40x (16kB) Flash CY8C50x (16kB) Flash CY8C50x Configuration CY8C50xx Configuration CY8C50xx CFG NVL | Flash Size 128k 256k 64k 128k 256k 512k 16k 32k 4B | Device Type 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 | Origin MDK Core | ^ |
| Description ATSAMR21 128kB Flash ATSAMR21 258kB Flash ATSAMR21 64kB Flash CC2538x 258 KB CC2538x 256 KB CC25500 Configuration CY8C5000 Flash CY8C5000 EEPROM | Flash Size 128k 256k 64k 128k 256k 512k 19k 32k 256k 32k 4B 2k | Device Type 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 | Origin MDK Core | ^ |
| Description ATSAMR21 128kB Rash ATSAMR21 258kB Rash ATSAMR21 258kB Rash CC2538xx 128 KB CC2538xx 256 KB CC2538xx 256 KB CC2538xx 256 KB CC2538xx 1648B Rash CY8C500x Configuration CY8C500x Configuration CY8C500x Fash Protection | Flash Size 128k 256k 64k 128k 256k 512k 10k 256k 512k 256k 32k 4B 256B | Device Type On-chip Flash On-chip Flash | Origin MDK Core | ^ |
| Description ATSAMR21 128kB Flash ATSAMR21 258kB Flash ATSAMR21 258kB Flash CC2538x 128 KB CC2538x 256 KB CC2538x 256 KB CC2538x 2512 KB CY8C500 (16kB) Flash CY8C500 Flash CY8C500 Flash CY8C500 Flash CY8C500 Flash CY8C500 Flash Protection CY8C500 WO NVL | Flash Size 128k 256k 64k 128k 256k 512k 16k 256k 32k 256k 32k 256k 4B 2k 2568 4B 2k 2568 | Device Type On-chip Flash On-chip Flash | Origin MDK Core | ^ |
| Description ATSAMR21 128kB Flash ATSAMR21 258kB Flash ATSAMR21 654kB Flash CC2538xx 128 KB CC2538xx 256 KB CC2538x 25 | Flash Size 128k 256k 64k 128k 256k 512k 16k 256k 32k 256k 32k 256k 32k 256k 256k 32k 256k 256k 32k 256k 2k 256k 2k 256k 28k 128k | Device Type On-chip Flash On-chip Flash | Origin MDK Core |] |
| Description ATSAMR21 128kB Rash ATSAMR21 258kB Rash CC2538x 128 KB CC2538x 256 KB CC2538x 256 KB CC2538x 256 KB CC2538x 512 KB CY8C50x (16kB) Rash CY8C50x Configuration CY8C50x Configuration CY8C50x CFG NVL CY8C50x Fash Protection CY8C50x Rash Protection CY8C50x WO NVL EFM32 Gecko/Tinv Gecko | Flash Size 128k 256k 64k 128k 256k 512k 32k 256k 32k 256k 32k 256k 4B 2k 256k 128k 256k 32k 256k 128k 256k 128k 28k 28k | Device Type On-chip Flash On-chip Flash | Origin 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

| KtProg2 CMSIS-DAP | - | ODE | Device Name | | Move |
|---------------------------|-----------|---------------|------------------|---|------|
| Serial No: 080C0583031054 | SWDIO 🖸 | Dx0BC11477 | ARM CoreSight SW | I-DP | Up |
| Firmware Version: 1.0 | | | | | Dow |
| | C Automat | in Detention | ID CODE: | | |
| I♥ SWJ Port: SW ▼ | C Manual | Configuration | Device Name: | | |
| Max Clock: 1MHz | Add | Delete Up | odate | AP | 0x00 |
| Debug | | | | | |
| Connact & Resat Ontions | - | | Cache Options | Download Options | - |
| connect a neset options | | | | and the second se | |