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

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

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

CY8CKIT-032

PSoC® 4 Analog Front End (AFE) Shield Kit Guide

Doc. #: 002-21814 Rev. **

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone (USA): +1.800.858.1810
Phone (Intnl): +1.408.943.2600
www.cypress.com

Copyrights

© Cypress Semiconductor Corporation, 2017. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC (“Cypress”). This document, including any software or firmware included or referenced in this document (“Software”), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress’s patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR 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, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.

Contents

Safety Information 5

1. Introduction 7

1.1 Kit Contents 8
1.2 Board Details 9
1.3 Kit and System Block Diagram 10
1.4 Getting Started 11
1.5 Additional Learning Resources 11
1.6 Technical Support 11
1.7 Document Conventions 12
1.8 Acronyms 12

2. Software Installation 13

2.1 Before You Begin 13
2.2 Install Software 13
2.2.1 PSoC Creator 13
2.2.2 WICED Studio 13

3. Kit Operation 14

3.1 Theory of Operation 14
3.1.1 Power Supply 14
3.1.2 PSoC 4 14
3.1.3 LEDs 15
3.1.4 Mechanical Buttons 15
3.1.5 SmartIO 16
3.1.6 CapSense Buttons 18
3.1.7 CapSense Proximity 18
3.1.8 ADC 18
3.1.9 Thermistor 19
3.1.10 Ambient Light Sensor 19
3.1.11 Potentiometer 19
3.1.12 Humidity 19
3.1.13 DAC Output 19
3.1.14 PSoC I2C Interface 20
3.1.15 U8G Graphics OLED Display 20
3.1.16 UART 20
3.1.17 Arduino Interface 21

4. Code Examples **22**
4.1 Shield Project 23
4.2 Bootloader Project 24
4.3 TestProgram4M Project 24
4.3.1 Test Procedure 24
4.3.2 Alternate Screens 26

5. Re-Programming the PSoC 4 on the Shield **27**
5.1 Programming 27
5.2 Bootloading 27

A. Appendix **29**
A.1 Hardware Design 29
A.2 Schematics 29
A.2.1 PSoC 4 Analog Coprocessor 29
A.2.2 OLED Display 30
A.2.3 Arduino Interface 30
A.2.4 User Interface (Buttons, LEDs, CapSense) 31
A.2.5 Analog Sensors 32
A.2.6 Power Supply 32
A.3 Bill of Materials 33

Revision History **35**

Safety Information

Regulatory Compliance

The CY8CKIT-032 Kit is intended for use as a development platform for hardware or software in a laboratory environment. The board is an open system design, which does not include a shielded enclosure. This may cause interference to other electrical or electronic devices in close proximity. In a domestic environment, this product may cause radio interference. In such cases, you may be required to take adequate preventive measures. In addition, this board should not be used near any medical equipment or RF devices.

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

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



The CY8CKIT-032 Kit contains electrostatic discharge (ESD) sensitive devices. Electrostatic charges readily accumulate on the human body and any equipment, and can discharge without detection. Permanent damage may occur on devices subjected to high-energy discharges. Proper ESD precautions are recommended to avoid performance degradation or loss of functionality. Store unused CY8CKIT-032 Kit boards in the protective shipping package.



End-of-Life/Product Recycling

This kit has an end-of life five years from the date of manufacture mentioned on the back of the box. Contact your nearest recycler for discarding the kit.

General Safety Instructions

ESD Protection

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

Handling Boards

The CY8CKIT-032 Kit board is sensitive to ESD. Hold the board only by its edges. After removing the board from its box, place it on a grounded, static-free surface. Use a conductive foam pad if available. Do not slide board over any surface.

1. Introduction

The CY8CKIT-032 is an Arduino Uno R3 compatible shield intended as a companion to add an analog front end (AFE) and user interface to a baseboard. Any baseboard with Arduino Uno R3 compatible headers and I^2C master functionality on the Arduino Uno R3 I^2C pins is compatible with this shield. Example compatible baseboards are the CYW943907AEVAL1F Wi-Fi kit, CYW920706WCDEVAL Bluetooth kit, and CY8CKIT-044 PSoC 4 M-Series Pioneer kit. A PSoC Creator project demonstrating the use of this shield with the CY8CKIT-044 PSoC 4 M-Series Pioneer kit is provided with the kit firmware.

A PSoC 4 Analog Coprocessor is used on the shield to read analog sensors (temperature, humidity, ambient light and potentiometer voltage) and to provide a CapSense user interface. The PSoC 4 acts as an I^2C slave so that the analog sensor values can be read by the baseboard.

The firmware pre-programmed into the PSoC 4 on the shield provides the AFE and user interface as described in this guide. The kit can be used with a compatible baseboard without re-programming the PSoC 4 on the shield. Instructions for reprogramming or bootloading new firmware is provided in Re-Programming the PSoC 4 on the Shield chapter on page 27 for users who want to modify the default shield behavior for some reason.

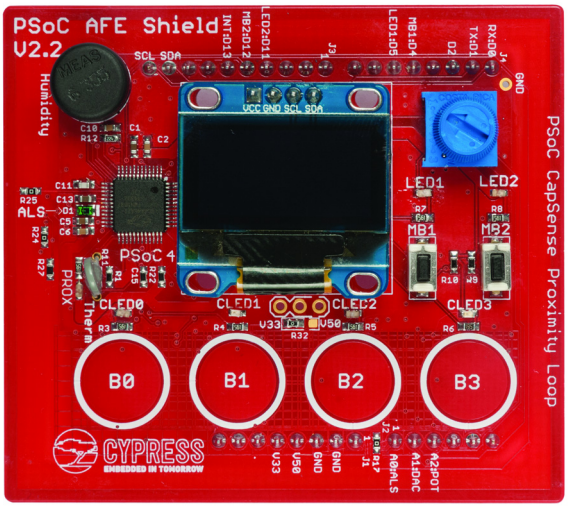
This kit is used in various Cypress Academy training classes and videos to facilitate teaching how to create devices for the Internet of Things (IoT). One example video class can be found at www.cypress.com/training/wicedwifi-101. That class shows how to use the shield with a CYW943907AEVAL1F Wi-Fi kit as the baseboard to create IoT devices based on the Wireless Internet Connectivity for Embedded Devices (WICED®) platform. Not all videos demonstrate the shield, but it is required for some of the exercises such as those using I^2C, the OLED display, and analog sensor measurements.

1.1 Kit Contents

The CY8CKIT-032 package has the following contents as shown in Figure 1-1.

- CY8CKIT-032 PSoC 4 Analog Front End (AFE) Shield
- Quick Start Guide

Figure 1-1. CY8CKIT-032 Board



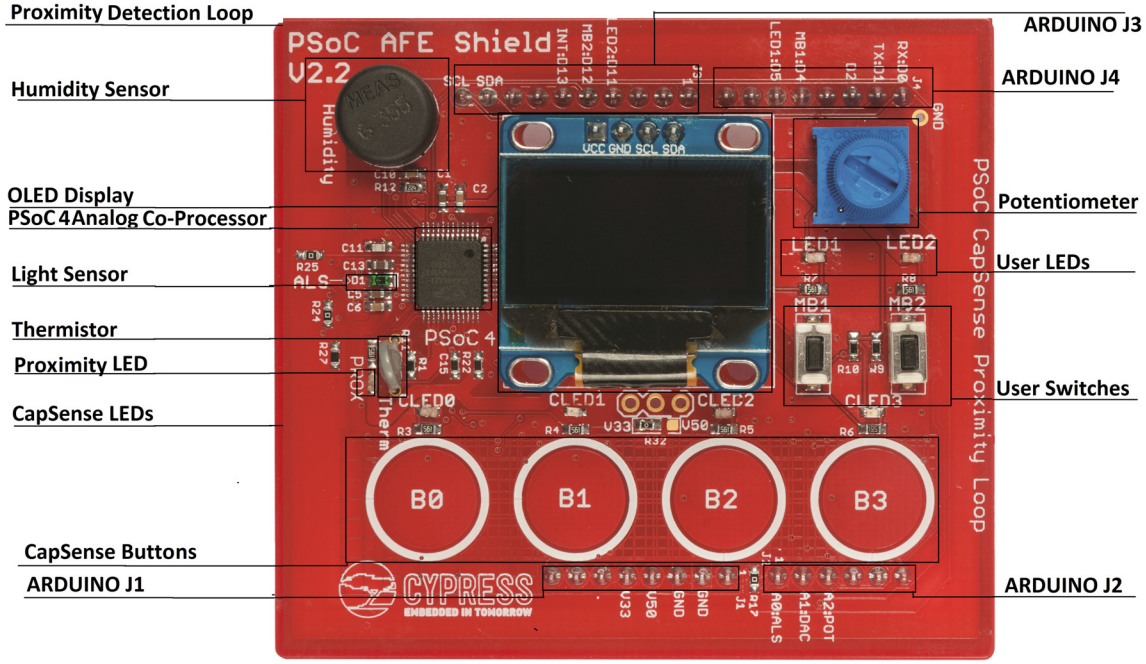
Inspect the contents of the kit; if you find any part missing, contact you nearest Cypress sales office for help or visit: www.cypress.com/support.

1.2 Board Details

The shield board has the following features:

1. PSoC 4 Analog Coprocessor MCU
2. Seven LEDs (2 User LEDs, 4 CapSense Button LEDs, 1 CapSense Proximity LED)
3. Two Mechanical Buttons
4. Four CapSense Buttons
5. Proximity Detection Loop
6. Thermistor
7. Ambient Light Sensor
8. Potentiometer
9. Humidity Sensor
10. 128 × 64 pixel graphics OLED Display
11. Arduino Compatible Headers
12. 5-Pin Programming Header (Behind the display, not populated)

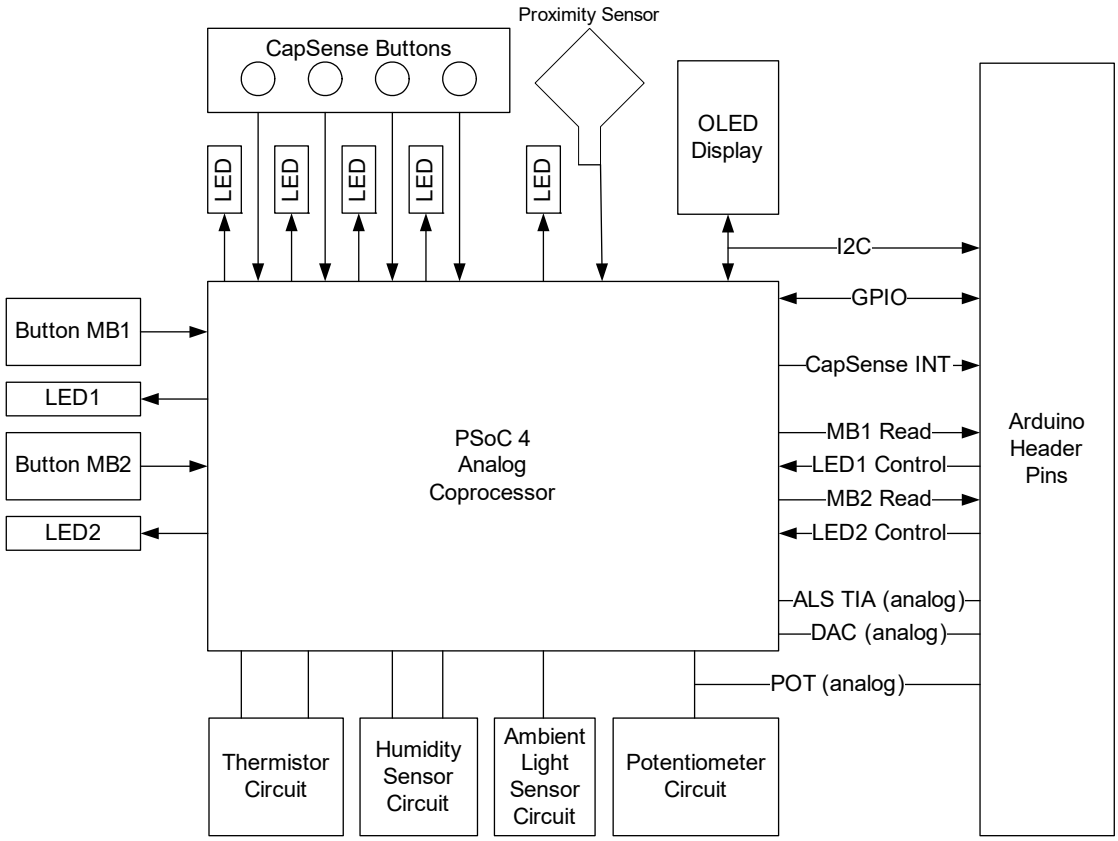
Figure 1-2. CY8CKIT-032 Peripherals



1.3 Kit and System Block Diagram

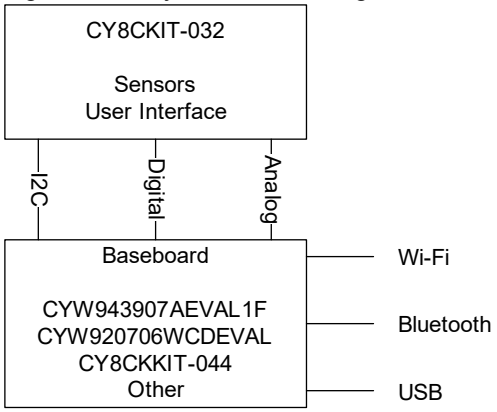
A block diagram of the shield is shown in Figure 1-3.

Figure 1-3. Block Diagram



A system block diagram showing the shield connected to an appropriate Arduino compatible baseboard is shown in Figure 1-4. The interface from the shield to the baseboard includes I^2C, Digital signals, and Analog signals as shown. The UART Tx and Rx from the PSoC 4 Analog Coprocessor are also connected to the Arduino UART pins, but the UART functionality is not implemented in the default firmware. The baseboard can interact with the outside world using WiFi, Bluetooth, USB, or a combination depending on what the baseboard supports.

Figure 1-4. System Block Diagram



1.4 Getting Started

This guide will help you to get acquainted with the CY8CKIT-032 shield kit:

- The Software Installation chapter on page 13 describes software that can be used to re-program the PSoC on the shield as well as software that can be used with various recommended baseboards.
- The Kit Operation chapter on page 14 describes the major features of the CY8CKIT-032 shield kit in detail.
- The Code Examples chapter on page 22 describes the PSoC project programmed by default into the PSoC on the shield as well as the bootloader project used to update firmware over I^2C and a test project that can be used with a CY8CKIT-044 PSoC 4 M-Series Pioneer kit to demonstrate all features of the shield.
- The Appendix chapter on page 29 provides a detailed hardware description, and the bill of materials (BOM).

1.5 Additional Learning Resources

A video class that demonstrates the use of this shield in an IoT device using the CYW943907AEVAL1F Wi-Fi kit as a baseboard is available at www.cypress.com/training/wicedwifi-101. Details on the kit used as a baseboard for the class can be found at www.cypress.com/CYW943907AEVAL1F.

1.6 Technical Support

For assistance, visit Cypress Support or contact customer support at +1(800) 541-4736 Ext. 3 (in the USA) or +1 (408) 943-2600 Ext. 3 (International).

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

- Cypress Developer Community
- Local Sales Office Locations

1.7 Document Conventions

Table 1-1. Document Conventions for Guides

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

1.8 Acronyms

Table 1-2. Acronyms Used in this Document

Acronym	Description
ADC	Analog-to-Digital Converter
AFE	Analog Front End
DAC	Digital to Analog Converter
ESD	Electrostatic Discharge
GPIO	General Purpose Input/Output
I^2C	Inter-Integrated Circuit
IoT	Internet of Things
LED	Light-Emitting Diode
LUT	Look Up Table
OLED	Organic Light Emitting Diode
PCB	Printed Circuit Board
POT	Potentiometer
PSoC	Programmable System-on-Chip
PWM	Pulse Width Modulation
TIA	Trans-Impedance Amplifier
UART	Universal Asynchronous Receiver Transmitter
VDAC	Voltage Output Digital to Analog Converter
WICED	Wireless Internet Connectivity for Embedded Devices

2. Software Installation

The PSoC 4 on the shield is pre-programmed with fully functional firmware that allows it to perform all functions described in this guide. If you want to re-program the PSoC 4 to implement your own custom features, then PSoC Creator 4.2 or later is required.

Depending on the baseboard used, software to program the baseboard will be required. For example, to use a WICED WiFi kit such as the CYW943907AEVAL1F kit as the baseboard, WICED Studio 6.0 or later is required. To use a PSoC kit such as the CY8CKIT-044 kit as the baseboard, PSoC Creator 4.2 or later is required.

2.1 Before You Begin

To install Cypress software, you will require administrator privileges. However, they are not required to run the software that is already installed. Before you install the kit software, close any other Cypress software that is currently running.

2.2 Install Software

2.2.1 PSoC Creator

If you want to re-program the PSoC 4 on the shield or use the shield with a PSoC kit as the baseboard, then you will need to install PSoC Creator 4.2 or later. That software is available for free from cypress.com/psoccreator. Follow the instructions on that page to install the tool and to find getting started information and training videos.

2.2.2 WICED Studio

If you want to use the shield with a WICED kit as the baseboard, then you will need to install WICED Studio. That software is available for free from cypress.com/products/wiced-software.

There is WICED WiFi video course at cypress.com/training/wicedwifi-101 that has detailed instructions on how to setup and use WICED Studio to work with WICED kits such as the CYW943907AEVAL1F. Platform files to use the shield with the CYW943907AEVAL1F kit are provided with the video course material. To use these files, copy the platform file folder into the WICED Studio SDK folder 43xxx_Wi-Fi/platforms/.

3. Kit Operation

This chapter introduces you to various features of the CY8CKIT-032 shield kit. The shield comes pre-programmed with firmware that implements all of the features described below. The firmware in the PSoC 4 on the shield does not need to be changed unless the user wants to modify the default behavior from what is described in this section.

3.1 Theory of Operation

3.1.1 Power Supply

The shield is powered from the Arduino power header pins. By default, the kit is powered by the 3.3 V Arduino pin. There is a 0-ohm resistor (R32) which can be moved to power from the 5.0 V Arduino pin instead. There is also an unpopulated 3-pin header (J5) which allows the power to be selected using a shunt instead of requiring moving the 0-ohm resistor to change the power supply pin. If J5 is used, then R32 must be removed.

See Power Supply on page 32 for the schematic of this section of the shield.

3.1.2 PSoC 4

The heart of the shield is a PSoC 4 (CY8C4A45AZI-483). This PSoC combines flexible Analog Front Ends, programmable Analog Filters, and high-resolution Analog-to-Digital converters along with an efficient-yet-powerful ARM® Cortex®-M0+ signal processing engine. The data from the PSoC can be accessed over I^2C, UART, or SPI. For this shield, I^2C is used as the default communication mechanism.

See PSoC 4 Analog Coprocessor on page 29 for the schematic of this section of the shield.

The PSoC 4 pin connections on the shield are indicated on the silkscreen on the back side of the shield.

The PSoC 4 is capable of sensing voltage, current, resistance, inductance and capacitance. For our purposes, we use:

- Resistance sensing for measuring temperature
- Current sensing for measuring ambient light
- Voltage sensing for measuring the voltage across a potentiometer
- Capacitance sensing for measuring humidity

3.1.3 LEDs

There are seven LEDs on the shield. Four are associated with the CapSense buttons (although they can be controlled independently via I^2C if desired as you will see later). One is associated with the proximity loop on the board – it turns on whenever proximity is detected. The other two LEDs are controlled by the baseboard. See User Interface (Buttons, LEDs, CapSense) on page 31 for the schematic of this section of the shield. All LEDs are active low, but the inversion is handled by the PSoC using the SmartIO block (see SmartIO on page 16) so that they appear to be active high to the WICED baseboard.

The two independently controlled LEDs are connected to the Arduino header as follows:

Table 3-1. LEDs connected to the Arduino header

LED	Header Pin
LED1	D5
LED2	D11

3.1.4 Mechanical Buttons

There are two mechanical buttons on the shield. These are active low and a pullup resistor is included for each of them on the shield. See User Interface (Buttons, LEDs, CapSense) on page 31 for the schematic of this section of the shield. They are connected to the Arduino header as follows:

Table 3-2. Mechanical Buttons connected to the Arduino header

Button	Header Pin
MB1	D4
MB2	D12

The state of the mechanical buttons can also be read via I^2C as you will see in PSoC I2C Interface on page 20.

3.1.5 SmartIO

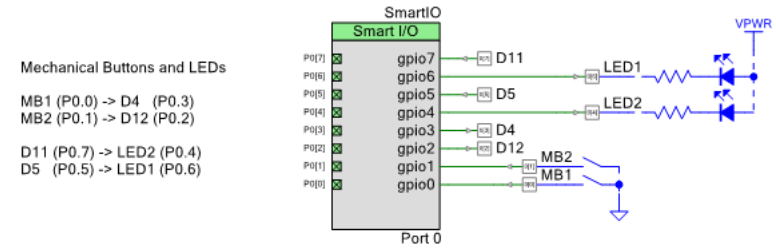
The two independent LEDs and the mechanical buttons are all controlled by the SmartIO block in PSoC 4. That is, rather than the LEDs and mechanical buttons connecting directly to the Arduino header pins, they are instead routed through the hardware SmartIO block in the PSoC. This is done for two reasons:

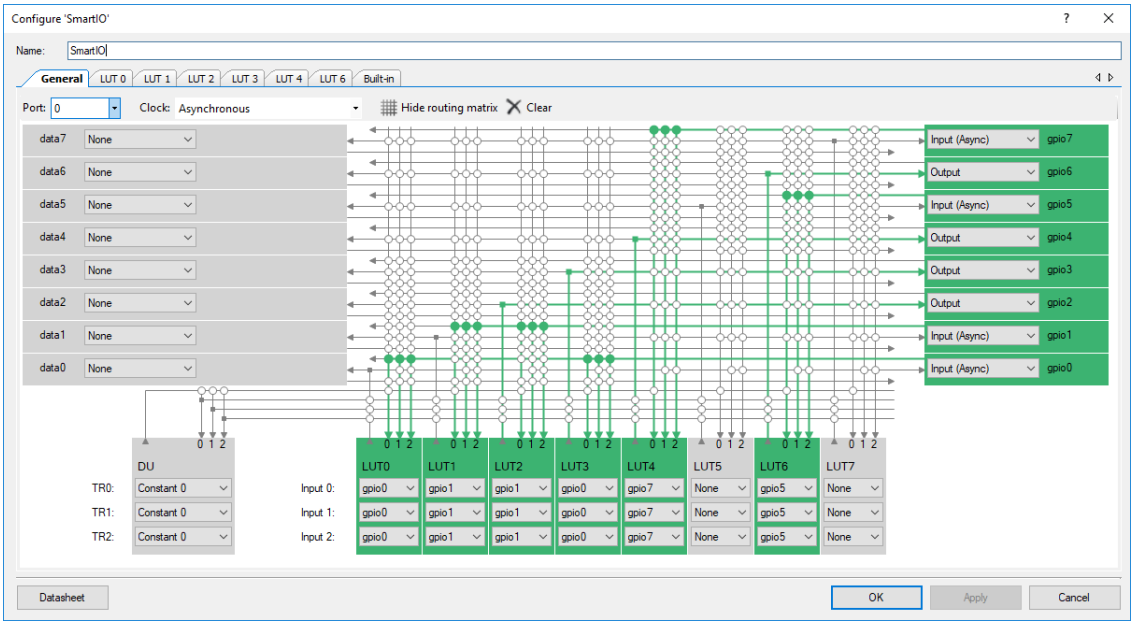
1. It allows the PSoC to control the resources directly when desired. For example, during bootloader mode, the LEDs are driven by the PSoC independent of what the baseboard does to the Arduino pins. This prevents possible contention.
2. It allows the PSoC to modify the apparent functionality of the hardware. For example, the LEDs are active low on the shield but the SmartIO block inverts the signals so that they appear to be active high to the baseboard.

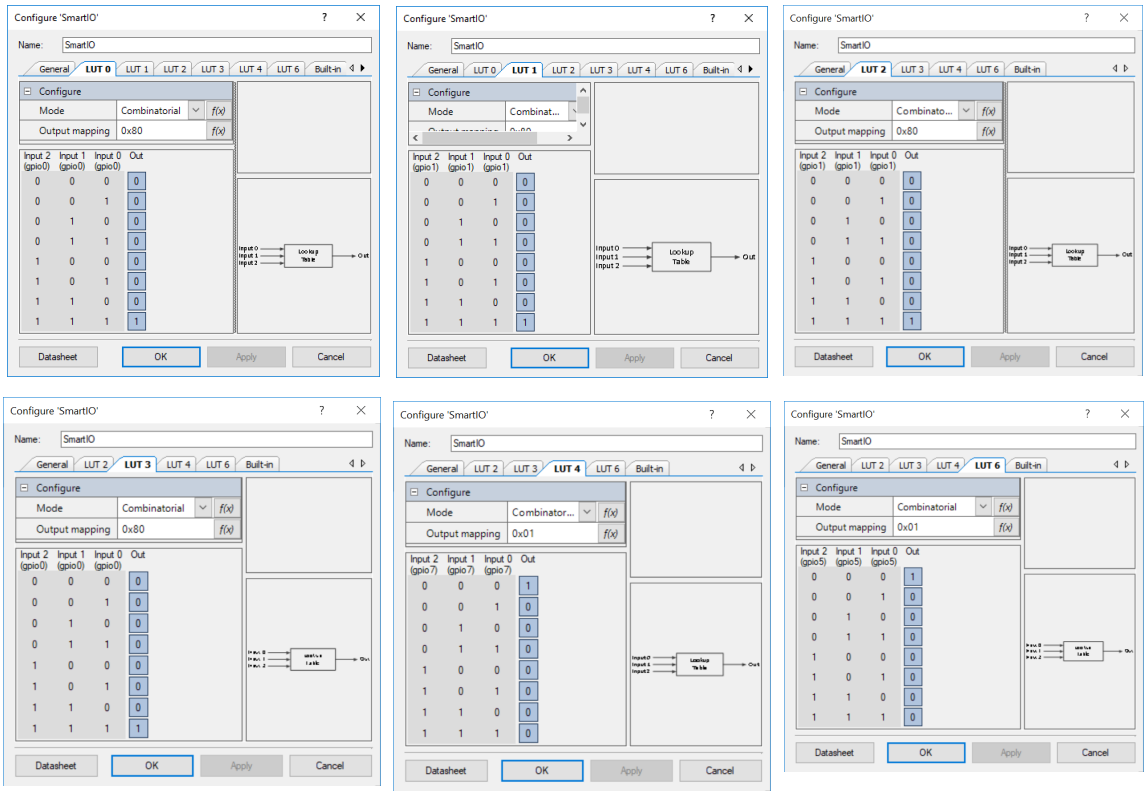
The schematic and the SmartIO configuration are shown in Figure 3-1. The connections are:

- MB1 routes to D4 using LUT3
- MB2 routes to D12 using LUT2
- D5 routes to LED1 and is inverted using LUT 6
- D11 routes to LED2 and is inverted using LUT4

Figure 3-1. Schematic and the SmartIO Configuration







Note: LUT0 and LUT1 are required so that the mechanical button input states can be read in the firmware.

3.1.6 CapSense Buttons

There are four CapSense buttons. These buttons are scanned by the PSoC and their state is reported via an I^{2}C register (see PSoC I2C Interface on page 20). See User Interface (Buttons, LEDs, CapSense) on page 31 for the schematic of this section of the shield.

By default, each CapSense button will light an LED when it is touched. The LEDs can be "decoupled" from the CapSense buttons if desired by setting bit 0 in the I^{2}C LED Control register. Once that is done, the LED Value register can be used to control the LEDs independently from the CapSense buttons.

There is also a dedicated active high CapSense interrupt line that goes high for 1–2 ms each time the state of one or more of the CapSense buttons changes. This signal is routed to Arduino header pin D13 so that the baseboard can use it if desired.

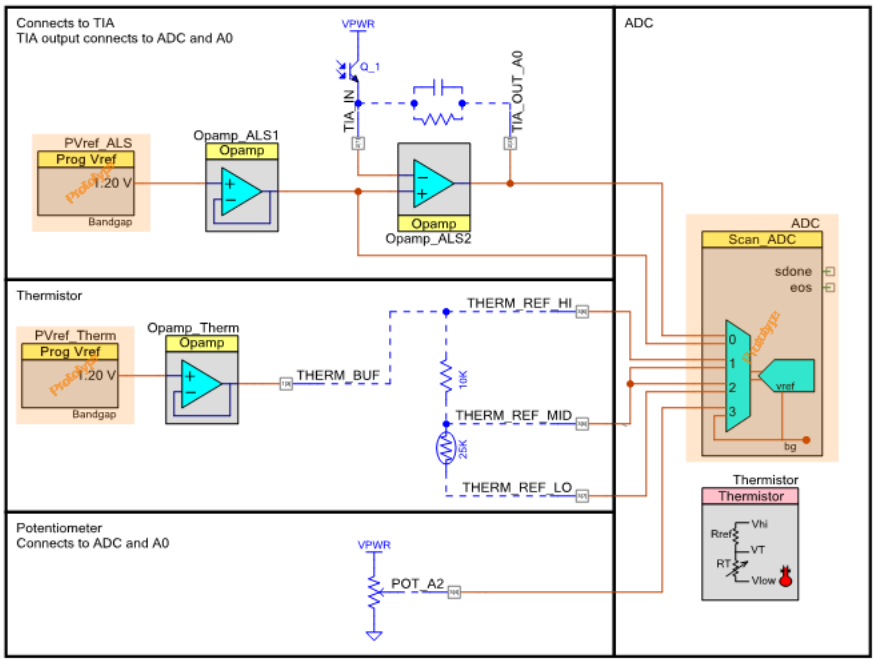
3.1.7 CapSense Proximity

There is a capacitive proximity sensor that runs around the outer edge of the board. The proximity sensor state is reported over I^{2}C (see PSoC I2C Interface on page 20). There is an LED dedicated to the proximity sensor – it turns on whenever an object is detected within ~5 cm of the board.

3.1.8 ADC

The PSoC Creator schematic for the Thermistor, Ambient Light Sensor, and Potentiometer analog system is shown below.

Figure 3-2. Sensor Configuration in PSoC Creator



3.1.9 Thermistor

The temperature is calculated by measuring voltage across a thermistor. See Analog Sensors on page 32 for the schematic of this section of the shield. The actual temperature calculation is handled by a PSoC component called “Thermistor” which greatly simplifies the coding required. The schematic and firmware are based on code example CE211321. The temperature value can be read over the I^2C interface (see PSoC I2C Interface on page 20) and is reported in degrees Celsius.

3.1.10 Ambient Light Sensor

The ambient light is calculated by measuring current through a photo-diode. See Analog Sensors on page 32 for the schematic of this section of the shield.The schematic and firmware are based on code example CE211252. The light value can be read over the I^2C interface (see PSoC I2C Interface on page 20) and is reported in LUX. In addition, the raw value at the output of the TIA can be measured at Arduino pin A0.

3.1.11 Potentiometer

The voltage across the potentiometer is measured and can be read over the I^2C interface (see PSoC I2C Interface on page 20) and is reported in Volts. See Analog Sensors on page 32 for the schematic of this section of the shield. The POT voltage will vary over the full power supply range but the ADC in the PSoC 4 is configured for a 0 V–2.4 V range so it will never report a value above 2.4 V. In addition, the raw POT voltage is available at Arduino pin A2.

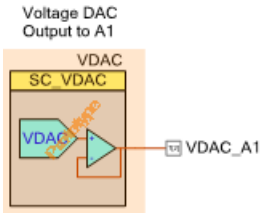
3.1.12 Humidity

Humidity is calculated by measuring capacitance of a humidity sensor using the CapSense block. See Analog Sensors on page 32 for the schematic of this section of the shield. The schematic and firmware are based on code example CE211322. The humidity value can be read over the I^2C interface (see PSoC I2C Interface on page 20) and is reported as a percentage.

3.1.13 DAC Output

A voltage DAC output is available on Arduino pin A1. The DAC voltage can be set in the range from 0V to 2.4V by setting a value in an I^2C register (see PSoC I2C Interface on page 20).

Figure 3-3. VDAC Configuration



3.1.14 PSoC I²C Interface

The PSoC I²C interface is an EZI2C slave. That is, the first byte of a write into the slave is an offset to the set of I²C registers. The remaining bytes (if any) are the data to be written starting at the offset. For I²C reads from the slave, the offset is whatever was set in the previous write. See the EzI²C component datasheet in PSoC Creator for additional details.

The I²C slave is assigned to 7-bit address 0x42 and is configured for a speed of 400 kbps. It is connected to Arduino pins SDA and SCL.

The I²C register map is as follows:

Table 3-3. I²C Register Map

Offset	Description	Format	Details
0x00	DAC Voltage	4 Byte Float	Desired DAC voltage in Volts
0x04	LED Value	1 Byte	CapSense LED values if LED Control bit 1 is 1. Mapping is: Bit 0: CLED0 Bit 1: CLED1 Bit2: CLED2 Bit3: CLED3
0x05	LED Control	1 Byte	Bit 0 sets how the 4 CapSense LEDs are controlled: 0 = CapSense Control 1 = Baseboard Control via the LED Value Register
0x06	Button State	1 Byte	State of CapSense, Proximity, and Mechanical Buttons: Bit 0: CapSense B0 Bit 1: CapSense B1 Bit 2: CapSense B2 Bit 3: CapSense B3 Bit 4: Mechanical Button 1 Bit 5: Mechanical Button 2 Bit 6: Proximity
0x07	Temperature	4 Byte Float	Temperature in °C
0x0B	Humidity	4 Byte Float	Humidity in %
0x0F	Ambient Light	4 Byte Float	Ambient light in lux
0x13	Potentiometer	4 Byte Float	Potentiometer in V

3.1.15 U8G Graphics OLED Display

The shield contains a U8G Graphics OLED display with an I²C interface. See OLED Display on page 30 for the schematic of this section of the shield. The OLED is an I²C Slave with an address of 0x3C which can be controlled from the baseboard using the I²C interface connected to Arduino pins D14 and D15.

3.1.16 UART

The Arduino

3.1.17 Arduino Interface

The Arduino pin functions are shown below. See Arduino Interface on page 30 for the schematic of this section of the shield.

Table 3-4. Arduino Pin Functions

Arduino	Shield Function
A0	Ambient Light TIA Output
A1	DAC Voltage
A2	Potentiometer
D0	UART RX to baseboard, UART TX from PSoC *
D1	UART TX from baseboard, UART RX to PSoC *
D2	Unused connection to PSoC
D4	Mechanical Button MB1
D5	LED1
D11	LED2
D12	Mechanical Button MB2
D13	CapSense Interrupt
SDA	I2C_SDA
SCL	I2C_SCL
RES	Reset (connected to PSoC XRES)
5 V	5 V power option (see Power Supply on page 14 for details)
3.3 V	3.3 V power option (default) (see Power Supply on page 14 for details)
GND	Ground

* UART functionality is not implemented in the default PSoC firmware.

A3, A4, A5, D3, D6, D7, D8, D9, D10, IOREF, AREF, and VIN are not connected on the shield.

4. Code Examples

A PSoC Creator workspace (called Shield.cywrk) is provided with the kit containing three projects.

Two of the projects (Shield and Bootloader) are intended for the PSoC 4 on the shield itself. The third project (TestProgram4M) is intended for a CY8CKIT-044 kit which can be used as a baseboard to demonstrate the functionality of the shield.

Note that any baseboard containing Arduino Uno R3 compatible headers and I^2C master functionality can be used as a baseboard - the CY8CKIT-044 project is just included as an example. The CY8CKIT-044 is not included with the shield.

Other kits which can be used as baseboards include the CYW943907AEVAL1F WICED WiFi Evaluation Kit and the CYW920706WCDEVAL WICED Bluetooth Evaluation Kit.

Details on the three projects included in the PSoC Creator workspace can be found in the following sections.

Note that the shield firmware described here is pre-programmed on the PSoC 4 on the shield and does not need to be modified or programmed onto the PSoC 4 to use the shield with a compatible Arduino baseboard. In fact, in most cases the shield will be used as-is. For training videos including complete examples for using the shield with the default firmware to create an IoT device using the WICED CYW943907AEVAL1F kit, see www.cypress.com/training/wicedwifi-101.

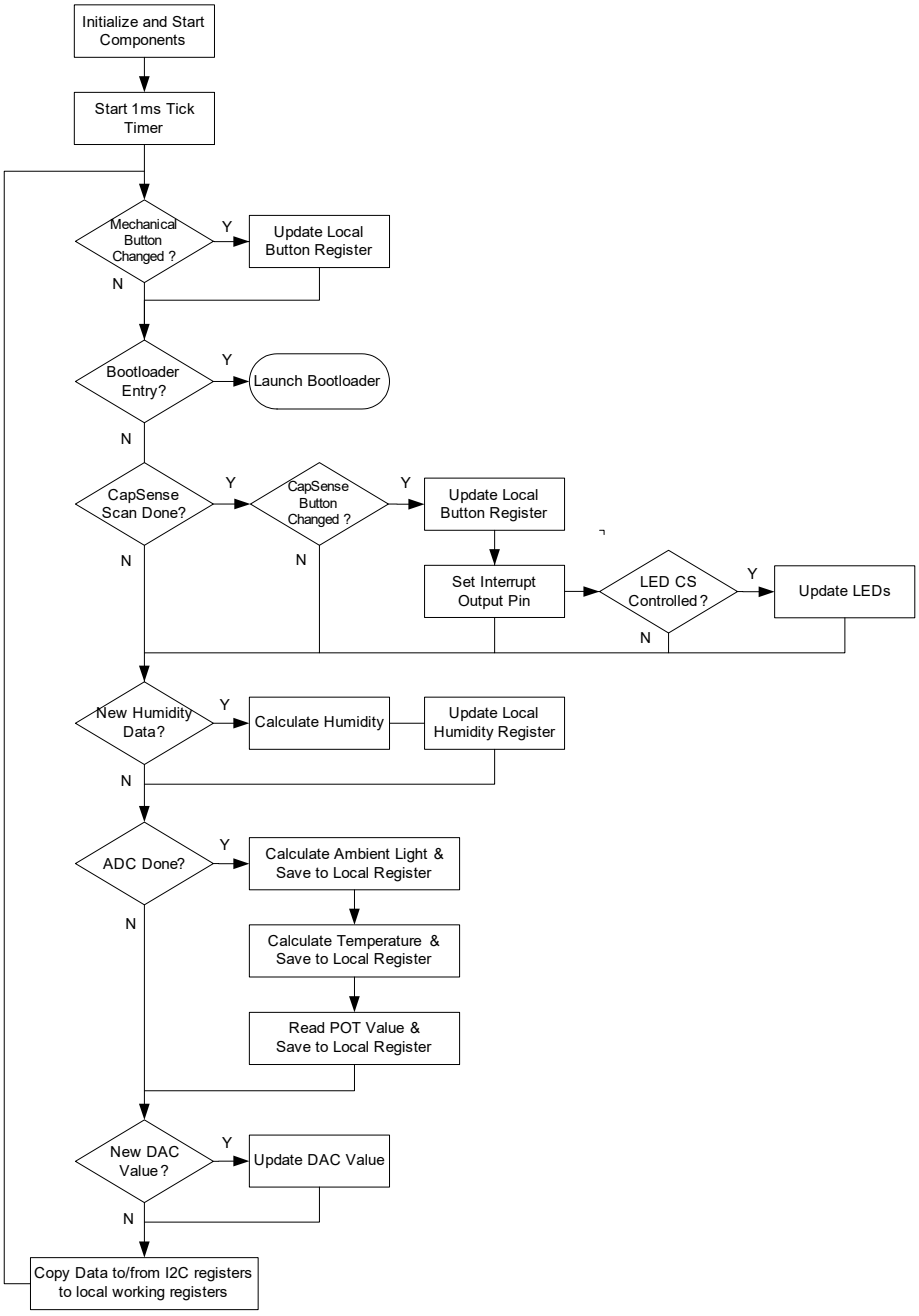
Table 4-1. Project Details

Project Name	Project Description
Shield	This project provides the default functionality described in the Kit Operation section. It is programmed into the board by default. There is no need to program this project to use the default kit functionality.
Bootloader	This project contains the bootloader that allows new firmware to be loaded using the I^2C interface. It is included as part of the shield project so that new firmware can be loaded without requiring a MiniProg3 connected to the 5-pin header. Instead, a baseboard containing an I^2C to USB bridge (such as the CY8CKIT-044) can be used to bootload the shield firmware. This project must be built before the Shield project can be built.
TestProgram4M	This project is a test program that can be programmed into a CY8CKIT-044 to be used as a baseboard to exercise the features of the shield. This is the default active project in the workspace.

4.1 Shield Project

The default shield project provides the features described in the Kit Operation chapter on page 14 of this guide. A flowchart of the firmware is Figure 4-1.

Figure 4-1. Flowchart



Since the Shield project references the Bootloader project, the Bootloader project must be built before building the Shield project.

The default active project in the PSoC Creator Workspace is the TestProgram4M project, so you must set the active project to Shield in PSoC Creator in order to build this project.

4.2 Bootloader Project

The bootloader project contains an I^2C slave interface and a bootloader component to enable bootloading of firmware to the PSoC 4. This project is referenced by the Shield project to add bootloader functionality so it must be built before building the Shield project. The default active project in the PSoC Creator Workspace is the TestProgram4M project, so you must set the active project to Bootlodaer in PSoC Creator in order to build this project.

The I^2C slave has a 7-bit address of 0x42 and is configured to operate up to 400 kbps.

To indicate bootloader mode, the project also blinks LED1 and LED2 in an alternating fashion at a frequency of 1 Hz.

The bootloader mode can only be exited by resetting the PSoC 4 on the shield board or by bootloading new firmware.

See Re-Programming the PSoC 4 on the Shield chapter on page 27 for details on bootloading new firmware to the shield.

4.3 TestProgram4M Project

The test project (TestProgram4M) can be programmed into a CY8CKIT-044 kit to demonstrate the functionality of the shield board. It can also be used for bootloading new firmware onto the Shield. Note that the CY8CKIT-044 is not included with the shield.

4.3.1 Test Procedure

At power-up, the OLED on the shield will display test information for each of the shield's features. Each feature will say Pass or Fail next to it depending on the test status. Some of the features are self-tested while others require user input. The test procedure is outlined below. Once all tests pass, a green LED on the baseboard will turn ON. The DAC, humidity, and temperature are self-tested so they should say Pass right away if the shield is operating properly.

Figure 4-2. Out of Box Program results using CY8CKIT-044

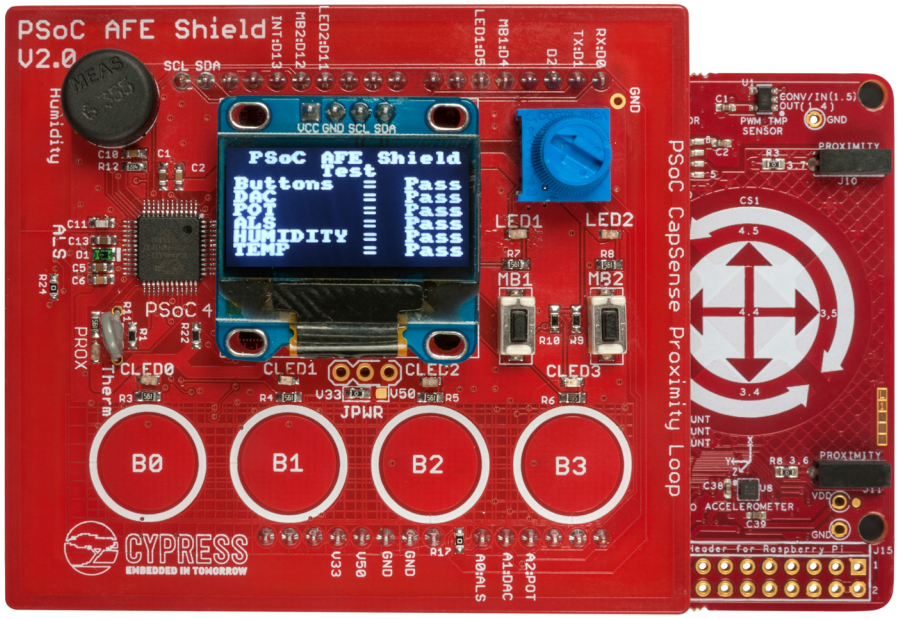


Table 4-2. Test Procedure

Feature	Method	Test Procedure
Buttons	Manual	Press each CapSense button and Mechanical button. An LED must turn on next to each button when it is pressed. Each button should be held down for approximately 0.5 sec to ensure that it is detected by the test program. Note that you can hold down more than one button at a time if you wish.
DAC	Automatic	The voltage is swept by the baseboard and the resulting voltage is measured on Arduino pin A1.
Potentiometer	Manual	Sweep the pot across its range. The voltage is measured on Arduino A2.
Ambient Light Sensor	Manual	Provide a wide range of light input to the sensor. Depending on environmental conditions you may need to cover the sensor with your hand or other object, and then shine a light directly on it to achieve the required range.
Humidity	Automatic	The humidity reading is examined for a valid result.
Temperature	Automatic	The temperature reading is examined for a valid result.

4.3.2 Alternate Screens

In addition to the main test screen, there are additional screens with more detailed information. Press and release user button SW2 on the baseboard to toggle between the following screens:

1. Main test screen: This is the main test results page as described above.
2. Analog Values Screen: Shows readings for temperature, humidity, illumination, potentiometer, Arduino pin A2 and Arduino pin A1[1]. Note that pin A1 is the DAC output which is continually swept by the test program in 100mV increments.
3. Base ADC Screen: Shows raw ADC readings in mV from A0, A1, and A2[2].
4. Buttons Screen: Shows real-time values for the CapSense buttons, proximity sensor, and mechanical buttons. The mapping is:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Unused	Proximity	MB2	MB1	B3	B2	B1	B0

The buttons screen also shows the LED value register and the LED control register, but these are not controlled by the test program so they should always read 0.

1. The ADC on the CY8CKIT-044 has a range of 0 - 2.048 V. Any value above the top end of the range will be displayed as 2.05 V.
2. The ADC on the CY8CKIT-044 has a range of 0 - 2.048 V. Any value above the top end of the range will be displayed as 2.05 V.

5. Re-Programming the PSoC 4 on the Shield

The firmware pre-programmed into the PSoC 4 on the shield provides the AFE and user interface as described in Kit Operation chapter on page 14. The shield can be used with a compatible baseboard without re-programming the PSoC 4 on the shield. Instructions for reprogramming or bootloading new firmware are provided in this section in case the user wants to modify the default shield behavior.

New firmware can be loaded using SWD programming or by bootloading over an I^2C interface. Each method is described below.

5.1 Programming

An unpopulated 5-pin programming header (J7) is provided to allow reprogramming of the PSoC 4 using a MiniProg3. Programming is done using PSoC Creator or PSoC Programmer software. See the PSoC Creator or PSoC Programmer web sites for additional information on using those tools to program the PSoC.

The programming header is located beneath the OLED so it must be accessed from the bottom of the board. By default, the PSoC 4 on the shield has a project containing an I^2C bootloader, so in most cases, the PSoC 4 firmware can be modified using the bootloader through a baseboard that contains a USB to I^2C bridge (such as the CY8CKIT-044). The programming header is only needed in cases where reprogramming is needed but the loaded firmware does not contain the bootloader code or does not provide a mechanism for bootloader entry.

5.2 Bootloading

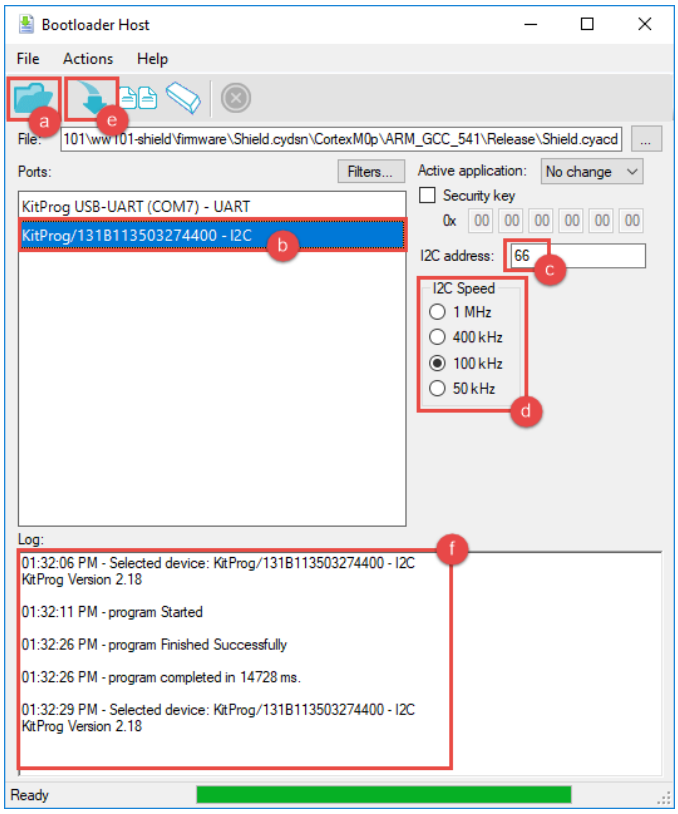
The firmware programmed into the PSoC 4 on the shield contains an I^2C bootloader. To bootload new firmware, a baseboard containing a USB to I^2C bridge which connects to the Arduino SCL and SDA pins is required. For example, the CY8CKIT-044 may be used as a USB to I^2C bridge.

The steps for using the CY8CKIT-044 kit to bootload new firmware to the PSoC 4 on the shield are as follows:

1. Create and build your desired firmware. The firmware must contain a “Bootloadable” component that references the bootloader project provided with the kit. See the “Shield” project for an example.
   a. Note that the Bootloader project must be built before building the Shield project so that the referenced files are available.
   b. Note that any custom firmware you create should provide a means to re-enter bootloader mode. If not, then you will not be able to bootload new firmware once your custom firmware has been loaded. At that point, you would have to reprogram the PSoC 4 using a MiniProg3 connected to the 5-pin programming header. See the “Shield” project for an example on entering bootloader mode.
2. Connect the shield to the CY8CKIT-044 and connect the kit to a PC using a USB cable.

3. The test project (TestProgram4M) utilizes the I^2C pins to display information onto the OLED display on the shield and to read values from the PSoC 4. Therefore, if the test project is loaded onto the CY8CKIT-044 kit, it is necessary to turn off those features to free up the I^2C interface for bootloading. To do that, press and hold button SW2 on the baseboard until LED3 on the baseboard blinks red at a frequency of ~ 5 Hz. This puts the CY8CKIT-044 into “bypass mode” so that the I^2C bus is not driven by the baseboard.
4. To put the PSoC 4 on the shield into bootloader mode, press and hold MB1 and MB2, and turn the POT through at least 1V of its range. Once bootloader mode starts, LED1 and LED2 on the shield will blink in an alternating fashion at a frequency of 1 Hz.
5. Open the Bootloader Host tool (available from the Tools menu in PSoC Creator) on your PC.
6. In the Bootloader Host tool (steps are illustrated in the figure below):
   a. Click the file button and provide the path to your project’s .cyacd file – this file can be found in the PSoC Creator project folder under: <ProjectName>.cydsn/CortexM0p/ARM_GCC_<Version>/Debug/<ProjectName>.cyacd
   b. Select the I^2C port for the kit.
   c. Set the I^2C address to 66 (0x42).
   d. Verify that the speed is set to 400 kHz or slower.
   e. Click the download button.
   f. View the log window to verify successful bootloading.

Figure 5-1. Bootloader Host Tool selection in Tools menu



7. Once bootloading has completed, press and release reset button SW1 on the CY8CKIT-044 kit to exit bypass mode.

A. Appendix

A.1 Hardware Design

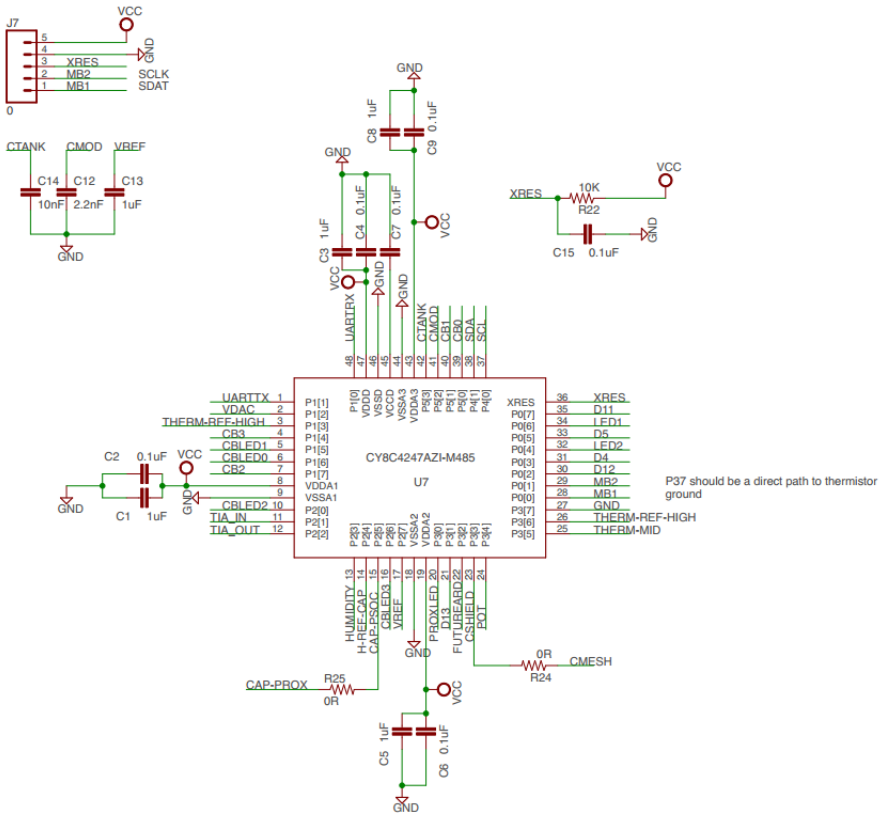
The hardware for this shield were done using the Eagle PCB design tools. The schematic and layout are available as part of the WICED WiFi 101 video training material which can be found at www.cypress.com/training/wicedwifi-101.

A.2 Schematics

A.2.1 PSoC 4 Analog Coprocessor

The PSoC 4 Analog Coprocessor is the analog front end and user interface to the baseboard. It includes a 5-pin programming header which can be used to re-program the PSoC if necessary, but most programming is done using the I^2C bootloader.

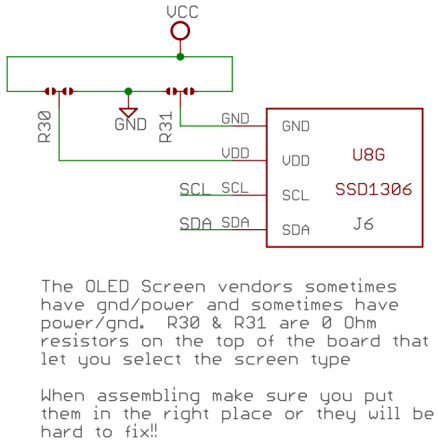
Figure A-1. PSoC 4 Analog Coprocessor Schematic



A.2.2 OLED Display

The OLED display is a complete module containing an SSD1306 controller and a 128x64 pixel dot-matrix OLED.

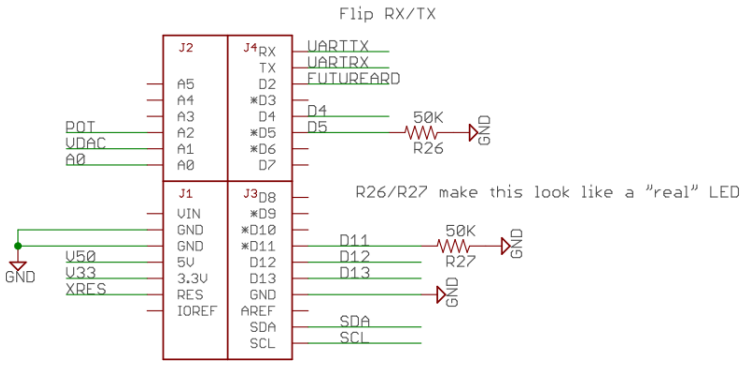
Figure A-2. OLED Display Schematic



A.2.3 Arduino Interface

A set of Arduino headers is provided for connection to a baseboard.

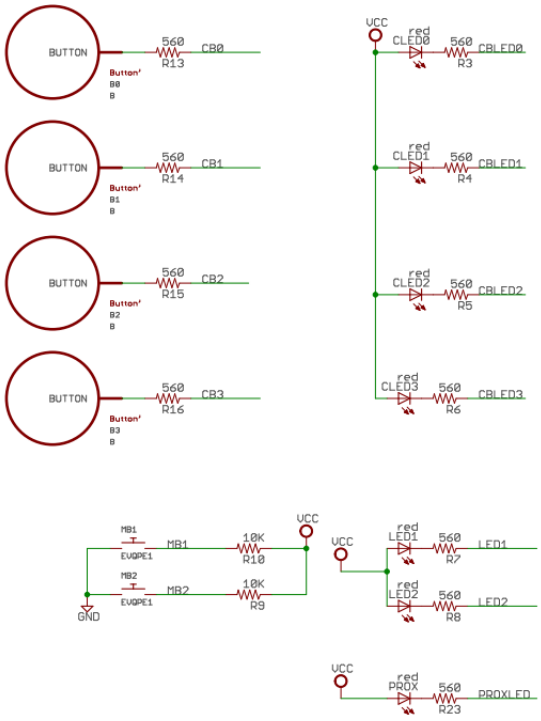
Figure A-3. Arduino Interface



A.2.4 User Interface (Buttons, LEDs, CapSense)

The user interface consists of 2 mechanical buttons, 4 CapSense buttons, 1 CapSense proximity loop, and 7 LEDs (one next to each button, and one for the proximity loop).

Figure A-4. Schematic of User Interface

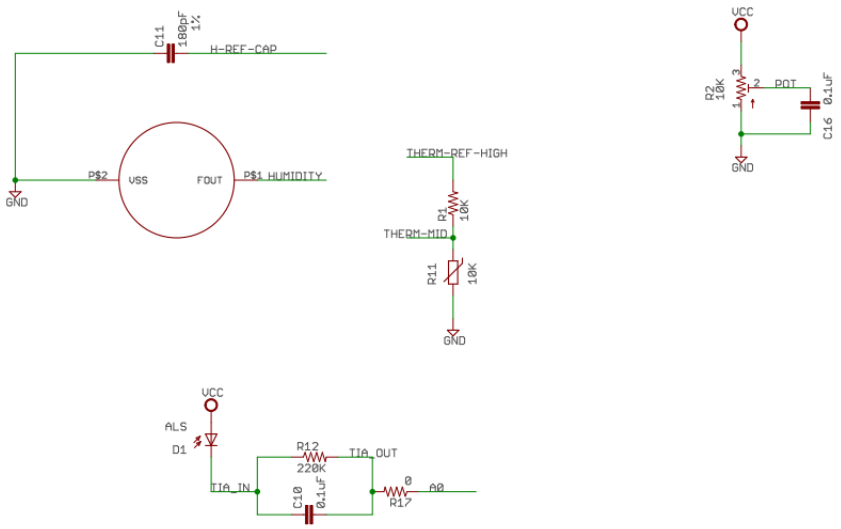


A.2.5 Analog Sensors

The analog sensors on the board are:

- Thermistor
- Ambient Light
- Humidity
- Potentiometer

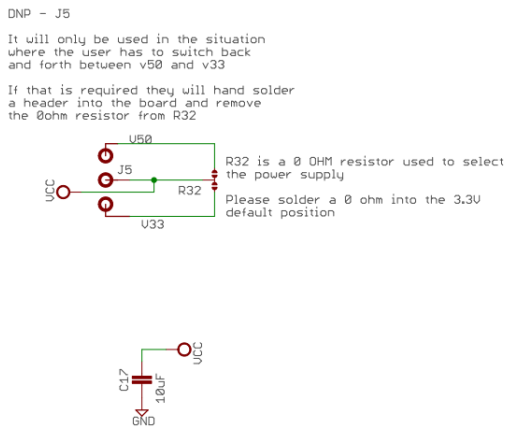
Figure A-5. Analog Sensor Interface



A.2.6 Power Supply

The power supply comes directly from the Arduino header pins. It can be powered either from the 3.3V pin (default) or the 5V pin. By default, a 0-ohm resistor must be changed to convert from 3.3V to 5V operation but there is an unpopulated 3-pin header that can be installed (and the 0-ohm resistor removed) to allow 3.3V / 5V selection by shorting the 2 appropriate pins using a shunt.

Figure A-6. Power Supply Schematic



A.3 Bill of Materials

Table A-1. Bill of Materials

#	Qty	Reference	Value	Description	Manufacturer	Mfr Part #
1	1	PCBA	N/A	PCB, 2.6 by 2.95 inch, Normal Tg, ENIG finish, 2 layer, Color = RED, Silk = WHITE	N/A	N/A
2	5	C13, C3, C8, C1, C5	1 uF	CAP CER 1 UF 16 V X5R 0603 ±10%	Murata	GRM188R61C105KA93D
3	8	C4, C9, C2, C6, C7, C10, C15, C16	0.1 uF	CAP CER 0.1 UF 16 V X7R 0603 ±10%	Murata	GRM188R71C104KA01D
4	1	C14	10 nF	CAP CER 10000 PF 50 V X7R 0603 ±10%	Murata	GRM188R71H103KA01D
5	1	C12	2.2 nF	CAP CER 2200 PF 50 V X7R 0603 ±10%	Murata	GRM188R71H222KA01D
6	1	C17	10 uF	CAP CER 10UF 25 V X5R 0603 ±10%	Murata	GRM188R61E106MA73D
7	1	C11	180 pF	CAP CER 180 PF 100 V C0G/NP0 0603 1%	Murata	GCM1885C2A181FA16D
8	1	H1	Sensor	SENSOR HUMIDITY 10 V CAPAC MOD	TE Connectivity	HPP801A031
9	4	R1, R9, R10, R22	10 kohm	RES SMD 10 KOHM 1% 1/10 W 0603 ±1%	Yageo	RC0603FR-0710KL
10	2	R26, R27	50 kohm	RES SMD 50 KOHM 5% 1/10 W 0603	Vishay Dale	TNPU060350K0HZEN00
11	1	R12	220 kohm	RES SMD 220 KOHM 5% 1/10 W 0603	Yageo	RC0603JR-07220KL
12	11	R13, R14, R15, R16, R3, R4, R5, R6, R7, R8, R23	560 ohm	RES SMD 560 OHM 5% 1/10 W 0603	Yageo	RC0603JR-07560RL
13	1	R2	10 kohm	TRIMMER 10 KOHM 0.5 W PC PIN	Bourns Inc	3386P-103TLF-ND
14	3	R24, R25, R17	0 ohm	RES SMD 0.0 OHM JUMPER 1/10 W 0603	Yageo	RC0603JR-070RL
15	3	R30, R31, R32	0 ohm	RES SMD 0.0 OHM JUMPER 1/16 W 0402	Yageo	RC0402JR-070RL
16	1	R11	Thermistor	NTC THERMISTOR 10 KOHM 1% BEAD	Murata	NXRT15XV103FA1B040

Table A-1. Bill of Materials *(continued)*

#	Qty	Reference	Value	Description	Manufacturer	Mfr Part #
17	1	Display	Display	Diymall 0.96" Inch Blue I^2C IIC Serial 128 x 64 Oled LCD LED Display Module for Arduino (pack of 2 pcs)	N/A	FZ1112-DIY
18	2	J1, J4	CON 8X1	CONN HEADER 8POS 0.100 13 mm length	Protectron	P9101-08-D32-1
19	1	J2	CON 6X1	CONN HEADER 6POS 0.100 13 mm length	Protectron	P9103-12-D32-1
20	1	J3	CON 10X1	CONN HEADER 10POS 0.100 13 mm length	Protectron	P9101-10-D32-1
21	2	MB1, MB2	Button	SWITCH TACTILE SPST-NO 0.05 A 12 V	Panasonic Electronic Components	EVQPE1
22	1	U7	Cypress IC	Cypress IC	Cypress Semiconductor	CY8C4A45AZI-483
23	1	D1	Light Detector Diode	AMBIENT LIGHT SENSOR 0805 SMD	Vishay Semiconductor Opto Division	TEMD6200FX01
24	7	CLED0, CLED1, CLED2, CLED3, LED1, LED2, PROX	Red LED	0603 - Red LED	Vishay Semiconductor Opto Division	TLMS1000
No Load						
1	1	J5	3 Pin	Three Terminal Jumper - Normally Open	Sulins	61300311121
2	1	J6	4 Pin	OLED Display header	Sulins	PPPC041LFBN-RC
3	1	J7	5 Pin	Programming Header	Sulins	640456-5
4	4	R18, R19, R20, R21	0 ohms	RES SMD 0.0 OHM JUMPER 1/10 W 0603	Yageo	RC0603JR-070RL

Revision History

Document Revision History

Document Title: CY8CKIT-032 PSoC® 4 Analog Front End (AFE) Shield Kit Guide Document Number: 002-21814				
Revision	**ECN#**	**Issue Date**	**Origin of Change**	**Description of Change**
**	5951300	12/21/2017	GJL	New kit guide.