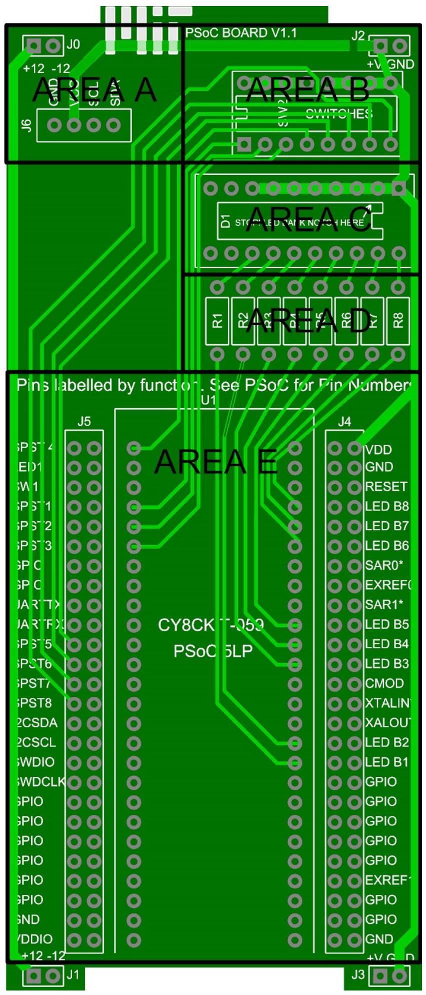
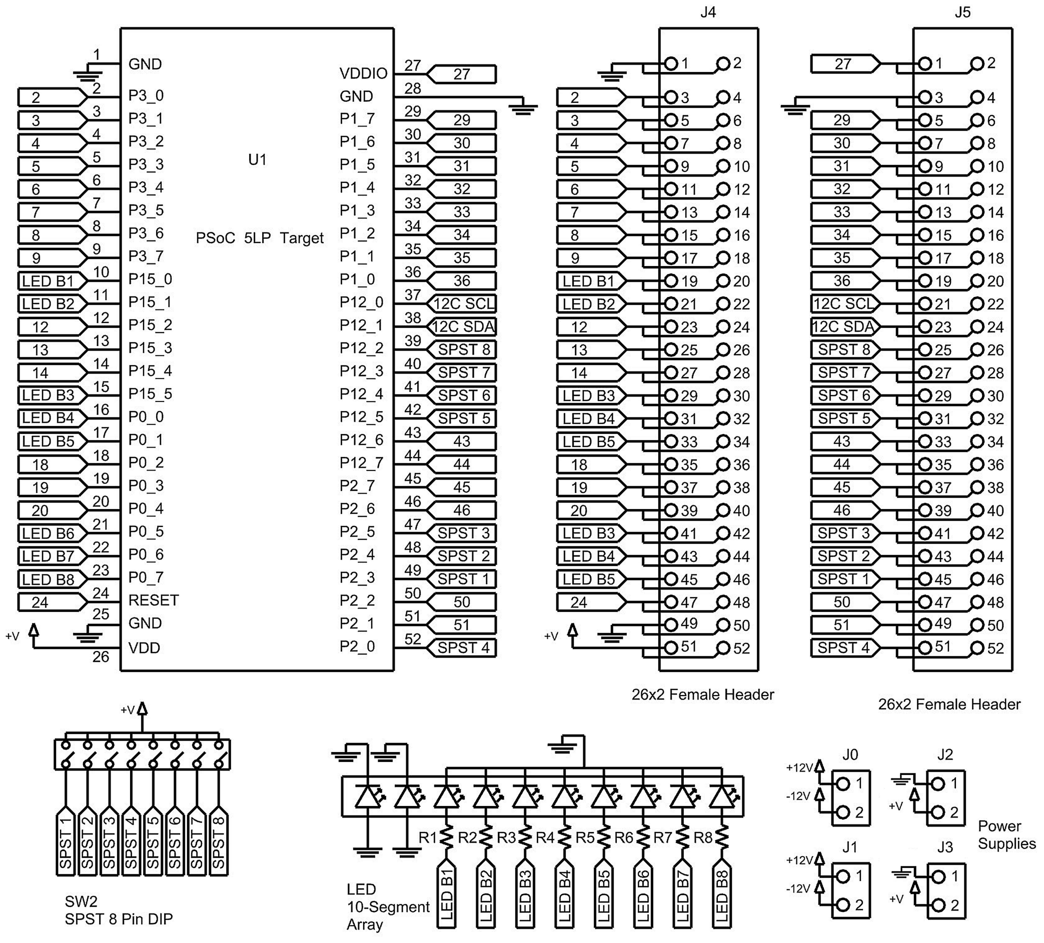
Massachusetts Institute of Technology
Electromechanical Systems Group

Electronics First: System-on-Chip

Board and **Circuit Schematic:**





Materials:

Part	Quantity	Part Number	Vendor	Part	Quantity	Part Number	Vendor
Breadboard	1	377 – 2646 – *ND*	DigiKey	*SW*2 - 8-Switch Slide DIP	1	*CT*2068 – *ND*	DigiKey
Breadboard Power Supply Module	1	-	-	*D*1 - 10-Segment LED Array	1	160 – 1068 – *ND*	Digikey
Multimeter	1	*MN*35 – *ND*	DigiKey	OLED Display	1	*B07F3KY8NF*	Amazon
2-pin Rail Connector Pins	4	952 – 2262 – *ND*	DigiKey	R_1-R_8 1 kΩ Resistor	8	*CF14JT1K00CT* – *ND*	DigiKey
Micro USB Power Cable	1	102 – 4123 – *ND*	DigiKey	16 Pin DIP Socket	1	*ED*3044 – 5 – *ND*	DigiKey
Soldering Iron and Solder Wire	1 ea	*T*0052918199*N* – *ND*	DigiKey	20 Pin DIP Socket	1	*ED*3069 – 5 – *ND*	DigiKey
*U*1 - PSoC 5LP CY8CKIT-059	1	*CY8CKIT* – 059	Cypress/Infineon	26*x*1 Male Connector Headers	2	399 – 4217 – *ND*	DigiKey
4*x*1 Female Connector Header	1	*S*7002 – *ND*	Digikey	26*x*1 Female Connector Headers	2	*S*7034 – *ND*	DigiKey
				26*x*2 Female Connector Header	2	*S*7129 – *ND*	DigiKey

The Build:

The Cypress/Infineon Programmable System-on-Chip (PSoC) is a powerful microcontroller with a wide variety of applications. This build will introduce you to the basic functionality and initial setup of this incredible device. You will see how the PSoC allows you to interact with and control user-oriented displays and input keys.

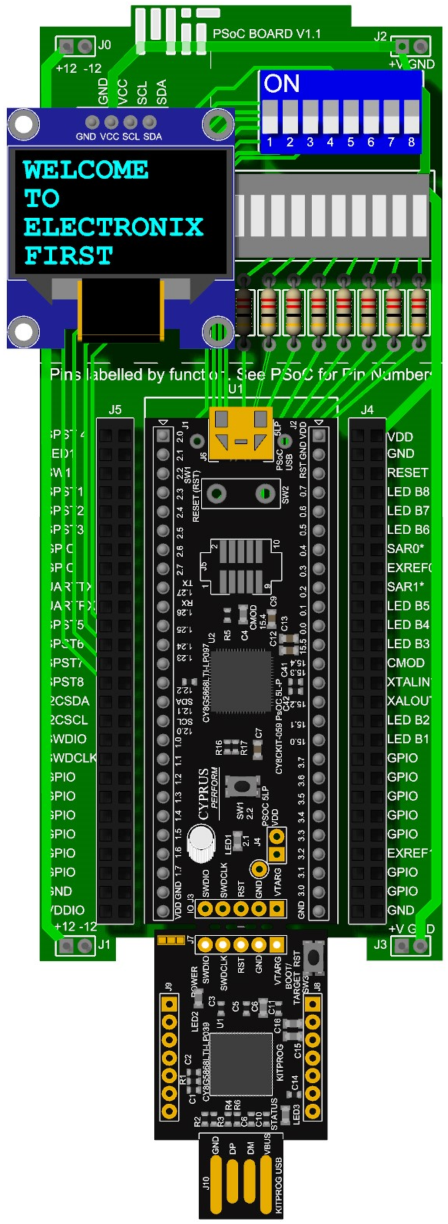
A MICROCONTROLLER IS ESSENTIALLY A COMPUTER ON A CHIP. A microcontroller can collect, store, and perform logical operations on input data. It can also produce output signals capable of controlling displays, audio devices, motors, and wide variety of other electromechanical components.

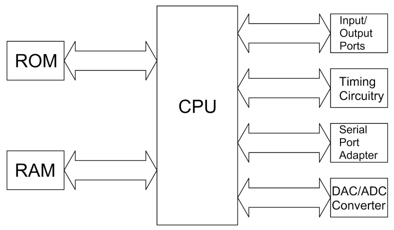
Many household electronic items that you interact with on a daily basis use these devices. Modern microwave ovens, for instance, completely rely on them. The keypad inputs, the LED display, the internal cooking clock, the heating elements can all be controlled from a single microcontroller.

What makes microcontrollers particularly useful in circuit design is that they have non-volatile internal memory that allows for the storage of instructions. Non-volatile memory, and thus the instructions, will persist even after rebooting the board. The instructions take the form of a written *program*, and they define the function of the microcontroller. The *program* is typically entered into the internal memory by way of programming software provided by the manufacturer.

The advantage of these devices is that, for many applications, a single properly-coded microcontroller can replace entire logic circuits. You have seen in previous labs how it takes a variety of individual components with very specific functions to complete a single complex circuit. Multiple pieces of hardware on the same board take up valuable real estate. If you could encapsulate a collection of circuitry in a single microcontroller, you could reduce the amount of physical space and cost required to complete a design. This sort of miniaturization represents a huge advantage in modern electronics. And, the functionality of a microcontroller circuit can often be updated in the field to improve or add functionality as new ideas or needs arise.

Most microcontrollers consist of the same basic components – a central processing unit (CPU), non-volatile Read-Only Memory (ROM), volatile Random Access Memory (RAM), input/output (I/O) ports, timing circuitry, interrupts, a serial port adapter, and some form of converter, be it digital-to-analog (DAC) or analog to digital (ADC). These components come together to provide an almost infinite set of possible functions for the microcontroller.


PSoC BOARD V1.1
ON
WELCOME TO ELECTRONIX FIRST
Pins labelled by function. See PSoC for Pin Number
CYPRESS PERFORM
The finished build.



Example of basic microcontroller components

At the heart or core of a microcontroller, a CPU reads and executes instructions programmed into non-volatile memory by the user. The CPU executes instructions following the periods or "beats" of an on-board clock. The instructions take the form of "ones and zeros" that directly correspond to actions the microcontroller will take, such as moving information in memory or adding numbers. Code is typically written in a human-readable form, such as the C programming language. It is compiled into the correct binary numbers and written into the non-volatile memory using software provided by the manufacurer. Volatile memory can be used to store temporary data created during the execution of those instructions. I/O ports connect the device to the external world.

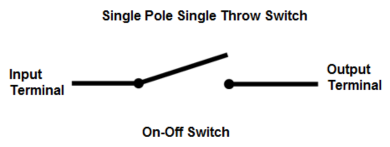
Theory of Operation and Predictions

This build consists of a set of switches, an LED array, an OLED display, and PSoC microcontroller circuit board. A programmable system-on-a-chip (PSoC) is a special type of microcontroller that allows for programmable hardware components in addition to software instructions. We will be using the CY8CKIT-059 ("PSoC stick"). The CY8CKIT-059 is a development board that contains a number of useful features connected to the PSoC. The PSoC stick has a USB connection allowing signals from a computer to be sent to the board. Code and hardware configurations can be uploaded via this connection into the internals of the PSoC microcontroller.

The DIP Switch

A dual inline package (DIP) switch is a collection of physically-actuated electrical switches aligned in a row. As you will see in this build, the geometry of the switch's pins typically allow for these kinds of switches to conveniently be inserted and removed from IC sockets which are directly wired to a circuit board. These switches serve as an on/off switch which can close or open a connection between the input and output. This type of switch is known as single-pole single-throw or SPST.

The primary purpose of the DIP switch is as user input. Each switch is connected to the microcontroller, and in our build, closing a switch applies a 5V voltage to a specific input. This allows us to manipulate the voltages on certain I/O pins and "communicate" with the device.



The LED Array

The 10-segment LED array is designed to display information in the form of illuminated bars. We however only use 8 of these LEDs with current-limiting resistors. This is an immediately recognizable way of displaying basic information non-numerically. They are most typically used in level-indicators, with the amount of LED bars illuminated indicating the approximate level of some measured parameter. This could be anything from the amount of fluid in a tank, to the volume of an audio signal being broadcasted. With the I/O pins, we are able to turn the LEDs on and off.

The OLED Screen

This build also includes an organic light-emitting diode (OLED) screen. An OLED is made of organic material that emits light when an electric current is applied to it. It functions similarly to LEDs you have seen in previous labs, however, it is much smaller, more efficient, and can be arrayed with other OLEDs to form a display. The end result is a screen with a relatively high resolution and the ability to display text and images which does not require a lot of space or computational power. As you might imagine, the practical applications of such a display are numerous. Though you'll be installing the display during the build, you won't be using it for this lab.

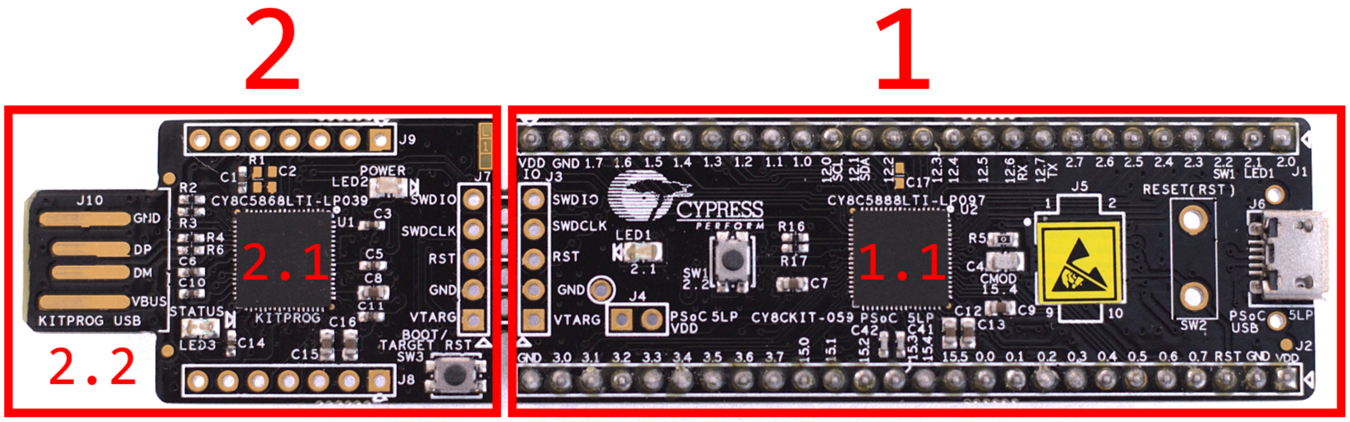
PSoC Functionality

The PSoC is a sophisticated microcontroller. It has a number of advantages over other similar devices, the most important of which is its advanced configurability. When designing a microcontroller-based system to solve a problem, most microcontrollers will require a set of physical external components (peripherals) to perform all the functions necessary for the system. These extra, purpose-built components take time to acquire when prototyping, and they can take up a lot of space in the final system. Even when a microcontroller comes with internal components, they may not be configurable and thus are more limited in their use.

The PSoC, on the other hand, contains in itself a suite of customizable digital and analog peripherals to suit a wide variety of applications. The configuration of these peripherals is stored in the controller's non-volatile memory, meaning it can be changed quite easily through its proprietary computer software, PSoC Creator. In essence, the PSoC contains "programmable hardware", allowing you to create functioning circuits using software tools without needing physical components, PCBs, or soldering. You can make connections from these programmed circuits to an outside system via the I/O pins on the PSoC, which are arranged by ports and pins. This means P1.7 corresponds to pin 7 of port 1.

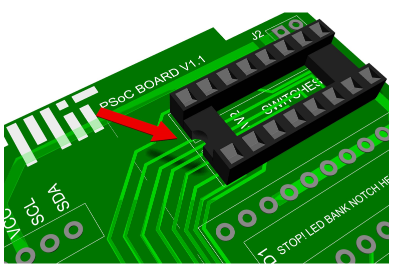
The wide array of configurable hardware components built into the PSoC gives this lab's build significant flexibility: you can perform a huge variety of tasks through this board alone. Experimentation is encouraged, but *please refrain from plugging in any external circuits without further guidance!* If you do not carefully understand what you are connecting to the PSoC you could very easily damage or destroy it.

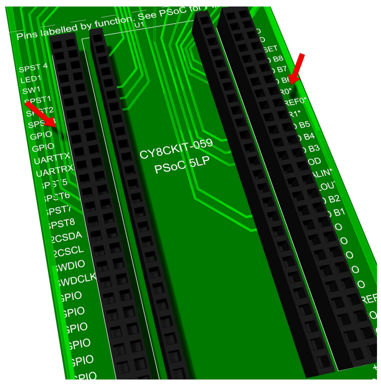
Anatomy of a PSoC Stick

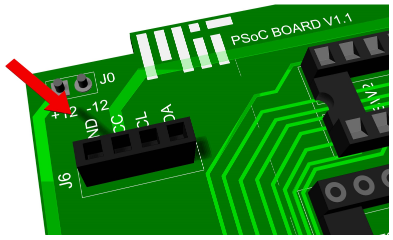


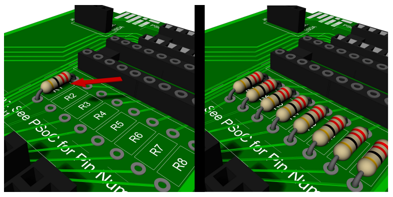
The picture above shows the PSoC stick you will be using. The PSoC itself is actually the tiny chip on the right side of the board (1.1); all of the neat PSoC abilities are contained on this integrated circuit (IC). The IC is mounted on the development board (1) created by Cypress/Infineon. The board makes it easy for developers (like you!) to work with the PSoC chip. The board also contains a few extra bells and whistles, such as a button, an LED, and a micro-USB port, all of which can be utilized in your own projects.

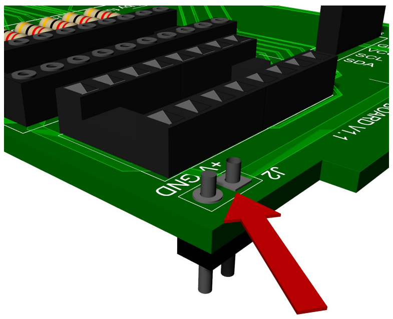
The left side of the stick (2) serves a slightly different purpose: it is used to program instructions into the program memory of the main PSoC chip. If you look closely, you may find that this part of the board contains a second PSoC chip (2.1)! This chip has already been programmed by Cypress/Infineon to program another PSoC chip (namely the one on the main board). It interfaces with the USB connector (2.2) and the PSoC Creator software, which you'll use later in this lab. In general, you don't have to understand how the programming process works in detail; just remember that when you program "the PSoC", you are programming the chip in the center of the board (1.1).


16-pin DIP Socket Installation


26x1 and 26x2 Female Connector Header Installation


4x1 Female Connector Header Installation


LED Array Fixed Resistors R_1-R_8 Location


Rail Connector Pin Installation

Assembly

For this build, alignment of the various header pins is critical. As such, you may be provided with a board with some of the components pre-soldered. If a component is already installed for you, move on to the subsequent step.

DIP Socket Installation

Install sockets but do NOT insert the associated components yet.

Install the 16-pin DIP socket for the 8-switch slide DIP in **AREA B**. Orientation matters - ensure the semi-circular notch at the top of the socket lines up with the corresponding notch illustrated on SW2 of the board. The pads are arranged as two rows of 8 contact points. Press the 16 socket pins through the corresponding pads on the board. Make good solder connections between each of the 16 socket pins and the metal board contacts located on the underside of the circuit board.

Repeat this process for the 10-segment LED Array 20-pin DIP socket in **AREA C**. Align, insert, and solder this socket into **D1**.

Board Headers

Install two 26x1 female connector headers in **AREA E**. Section **U1** contains two vertical columns of 26 contact points, one on the left and one on the right. *Alignment of these headers is critical, use caution*. Insert the header pins into the contacts and make soldered joints on the underside of the board. Repeat this process for the the two 26x2 female connector headers at **J4** and **J5**.

Next, install one 4x1 female connector header in **AREA A**. Locate the 4 contact points at **J6** and properly insert and solder the header pins into the board.

Resistor Installation

Install the LED Array fixed resistors in **AREA D**. These are current-limiting resistors which protect the LEDs in the 10-segment array. Insert each pair of resistor pins into the corresponding pair of holes at R_1-R_8. Make good soldered connections on the opposite side of the board and trim the excess leads.

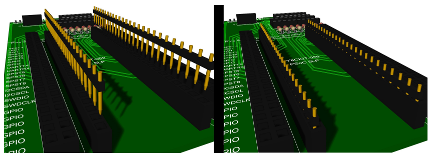
Breadboard Headers

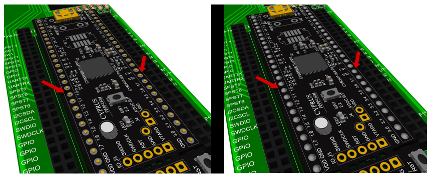
Install 2-pin rail connector headers at **J0**,**J1**,**J2**, and **J3**. These are installed with the base of the pin on the underside of the board, and the location of the soldered joint on the top of the board. ***BEFORE SOLDERING***, insert the bottom of the pins into your breadboard, and *then* make your soldered connections on the circuit board to ensure proper alignment.

PSoC 5LP Development Kit Board

Insert the two 26*x*1 male connector headers in the corresponding female headers you installed at **U1** on your main board. Push the long header pins until they are properly seated in the sockets. This will ensure proper alignment.

Then, locate the two vertical columns of 26 contact points on your black PSoC 5LP development board - one at **J1** on the left and one at **J2** on the right. Push the shorter pins at the top of the header up through the contact points from the underside of the board. Make good soldered joints at the contacts on the top of the board. *Be very careful not to allow solder to flow onto the board and nearby components!*

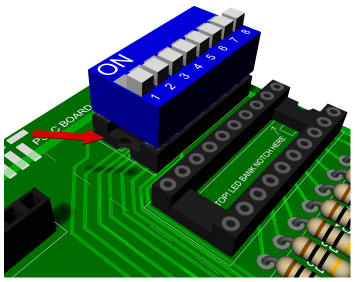

26*x*1 Male Connector Header Pin Installation

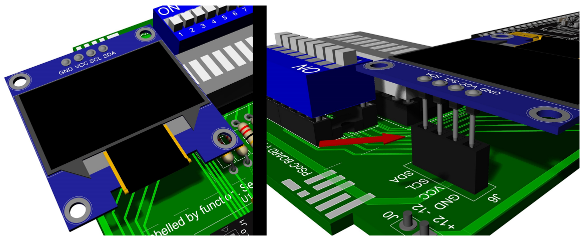

PSoC 5LP Development Board Header Installation

Integrated Circuit Installations

Once all of your soldered joints are completely cooled, carefully install the 8-switch slide DIP and the 10-segment LED Array into their respective sockets at **SW2** and **D1**. Be sure to put the correct IC into the correct socket! *Orientation matters!* Ensure that the notch on the top side of each component body aligns with the notch on the corresponding DIP socket.

Finally, install your OLED screen at **J6**. Insert the four connector pins on the underside of the screen into the four corresponding sockets in female header.

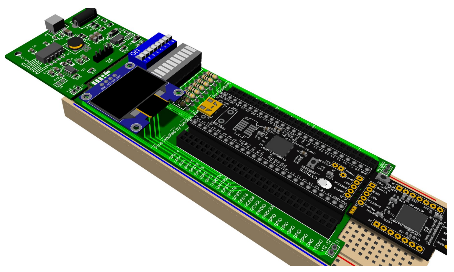

8-Switch Slide DIP Installation



OLED Screen Installation

Final Steps

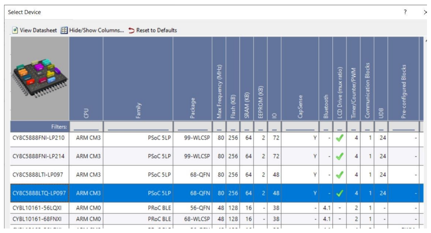
Carefully connect your circuit board to the breadboard in line with the power supply module. Ensure that your USB power cord is properly fitted into the micro USB port on the power supply module. Check that its indicator lights are illuminated. Also, ensure that the pins of both boards are fully inserted into the breadboard contact holes. Finally, disconnect the breadboard from power and continue.


Board and Power Supply Positioning

Setup

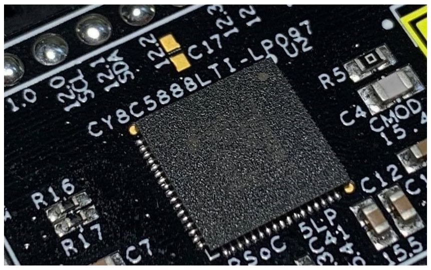
To properly access the PSoC functionality, you will first need to acquire some software. The specific tool you are looking for is called “Creator”, and it can be found at **web.mit.edu/cdev**. Download the PSoC Creator installation file to install PSoC Creator 3.3. Follow the installation instructions. This piece of software will allow you to easily upload code into the PSoC microcontroller. Programming software such as Creator handle the process of converting your code and schematics into a format (binary) that the PSoC can actually understand and store in its non-volatile memory.

You can find more versions and other reference material at **https://www.cypress.com/products/psoc-creator-integrated-design-environment-ide**. For example, you will also find a "PSoC Creator User Guide" available for free download. It is highly recommended that you get this guide and use it liberally as a reference. It also includes basic tutorials and example projects with which you can experiment.



	CPU	Family	Package	Max Frequency (MHz)	Flash (KB)	SRAM (KB)	EEPROM (KB)	IO	CapSense	Bluetooth	LCD Drive (mux ratio)	Timer/Counter/PWM	Communication Blocks	UDB	Pre-configured Blocks
Filters:															
CY8C5888FNI-LP210	ARM CM3	PSoC 5LP	99-WLCSP	80	256	64	2	72	Y	-	✓	4	1	24	-
CY8C5888FNI-LP214	ARM CM3	PSoC 5LP	99-WLCSP	80	256	64	2	72	Y	-	✓	4	1	24	-
CY8C5888LTI-LP097	ARM CM3	PSoC 5LP	68-QFN	80	256	64	2	48	Y	-	✓	4	1	24	-
CY8C5888LTQ-LP097	ARM CM3	PSoC 5LP	68-QFN	80	256	64	2	48	Y	-	✓	4	1	24	-
CYBL10161-56LQXI	ARM CM0	PRoC BLE	56-QFN	48	128	16	-	38	-	4.1	-	2	1	-	-
CYBL10161-68FNXI	ARM CM0	PRoC BLE	68-WLCSP	48	128	16	-	38	-	4.1	-	2	1	-	-

Device Select Menu



The device number on the stick

LED Exercise

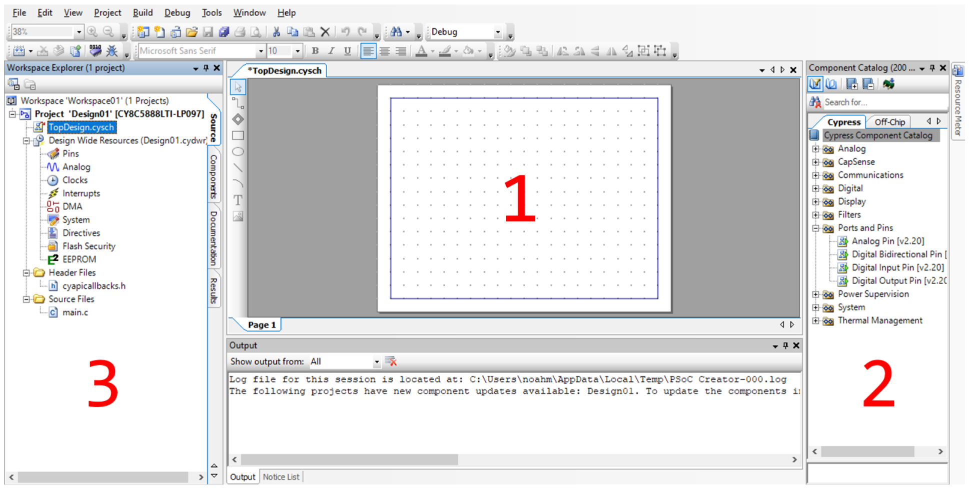
Now that your PSoC board is set up, you will do an exercise to get familiar with the PSoC Creator software. The software, used to program your PSoC chip, is both incredibly powerful and somewhat complex. To start, you will use the parts of Creator relevant to the PSoC’s programmable hardware to control the LEDs on your board with the DIP switches.

Creating a Workspace/Project

First, you will need a place to put your work. In the top left of Creator, click *File* > *New* > *Project*. You will need to select the target device for the project. Check *Target Device*, then select *Launch Device Selector* in the dropdown menu. Find $CY8C5888LTI - LP097$, or the device number written on your PSoC stick just above the central chip, and click *OK*. Select the *Empty Schematic* option, and then provide an appropriate workspace name (such as "Lab PSoC") and project name (such as "LED Exercise"). Be sure to save the workspace where you can find it later.

A Brief Creator Tour

With your workspace and project created, Creator should now look something like below. In the middle is the design schematic (1), also known as *Top Design*. This is where programmable hardware components will go; available components are found in the *Component Manager* (2). On the left is the *Workspace Explorer* (3), which shows your workspace, the projects in the workspace, and all the resources and files associated with a given project.



```
Log file for this session is located at: C:\Users\noahm\AppData\Local\Temp\PSoC Creator-000.log
The following projects have new component updates available: Design01. To update the components i
```

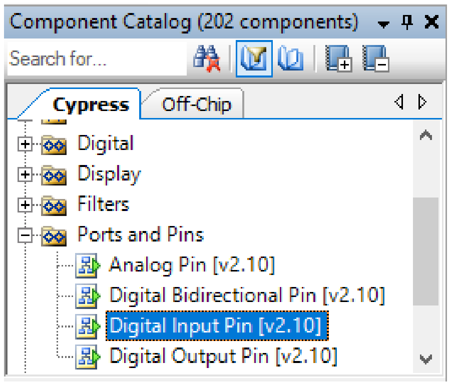
Adding Input Pins

For the PSoC to receive the status of the switches on the 8-switch slide DIP, it needs input pins. If it isn't open already, open *Top Design* by double clicking it in the *Workspace Explorer*. You should see a blank grid. To get an input pin, either search "pin" in the *Component Catalog* search bar or go to *Ports and Pins* > *Digital Input Pin*. Drag the listed component from the catalog window onto the schematic. Double click the component to open its configuration menu. It is recommended to change the pin's name to something like *SPST*1, since this is how it is labeled on your board.

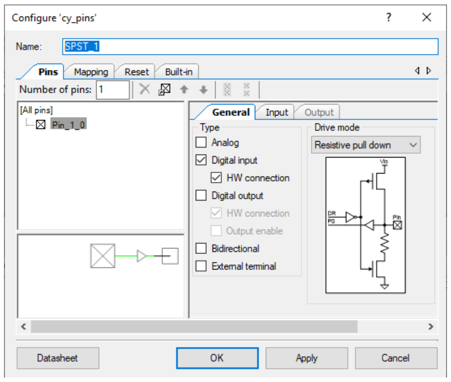
Input pins read voltages applied to them. They can be either logic LOW (0), or logic HIGH (1), and they can be connected to other programmed circuits in the PSoC or read directly by user code.

There is one small problem with our pin: when the switch connected to the pin is closed, it will apply 5V (HIGH) to the pin as expected; however, when the switch is open, the input pin is disconnected from the external input, and nothing is forcing the pin's voltage down to 0V (LOW). This leaves the pin in an unpredictable "floating" state, which could cause the pin to read HIGH even when the switch is open. We can fix this using the PSoC's configurability: in the pin's configuration menu, set the pin's drive mode to *Resistive pull down*. Now, when the switch is open, the pin is internally connected to a resistor that keeps it in the LOW state.

Make a total of 8 such input pins. The easiest way to do this is to configure one pin, then select it, copy it, and paste it in *Top Design* as many times as needed.

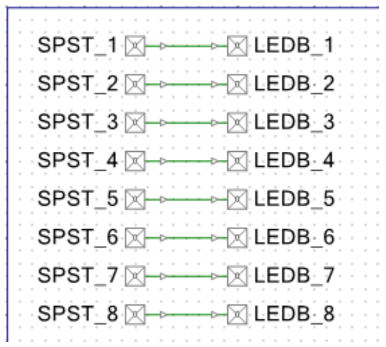

Component Catalog (202 components)
Search for...
Cypress
Off-Chip
Digital
Display
Filters
Ports and Pins
Analog Pin [v2.10]
Digital Bidirectional Pin [v2.10]
Digital Input Pin [v2.10]
Digital Output Pin [v2.10]

Creating an Input Pin



Name: SPST_1
Number of pins: 1
Type: [ ] Analog; [✓] Digital input; [✓] HW connection; [ ] Digital output; [✓] HW connection; [ ] Output enable; [ ] Bidirectional; [ ] External terminal
Drive mode: Resistive pull down

Input Pin Setup



Fully Connected Pin Setup

Adding Output Pins

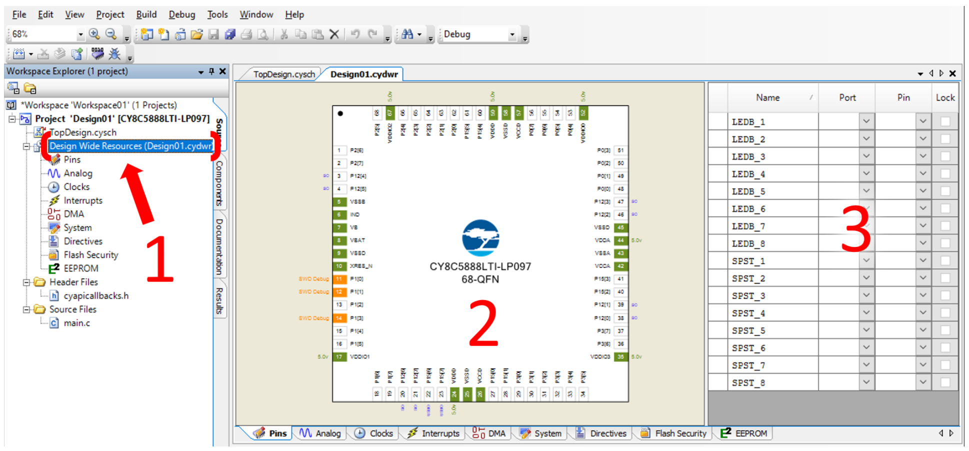
For the PSoC to send the state of the DIP switches to the LEDs, it needs output pins. Like before, find a digital output pin in the *Component Catalog*. Name it *LEDB*_1, since this is how the LED outputs are named on the board. The output pins should be set to strong drive, which connects the pin directly to either 0V or 5V instead of going through a large resistance (as it would in other modes). This enables us to brightly light up the LEDs since they already have a current-limiting resistor. Like the input pins, output pins can be connected to programmed circuits in the PSoC or can be driven by software.

Make 8 output pins, one for each input. To connect the inputs to the outputs, select the Wire Tool on the left side of *Top Design* (or press W on your keyboard), click the end of Input 1, and then click the end of Output 1. Draw similar wires between each input and output pair.

Connecting to Physical Hardware

Creating a schematic isn't quite the end of the story. The input and output pins in *Top Design* aren't yet connected to the physical pins on your PSoC. You can see the physical pins on your board: the pins you will use for inputs are labeled *SPST* with a number after, while the pins for outputs are labeled *LED B* with a number after. The PSoC identifies physical pins by a number written on the stick next to the pin. For example, *SPST*1 is pin 2.3 to the PSoC. To save your eyes some strain, the next page contains a handy listing of all the pins.

You will use Creator to connect to the physical pins. In the *Workspace Explorer*, double click on *Design Wide Resources* (1). You should now see something like this:



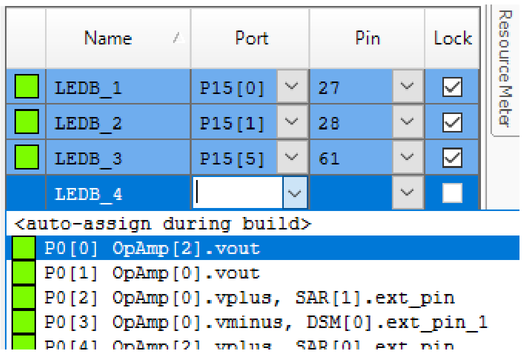
In the center is a graphic of the PSoC chip and all its pins (2). This corresponds to the chip in the center of the PSoC stick. On the right, you will find a list of the input and output pins you added in your *Top Design* schematic (3). Whenever you add a pin component to your schematic, it will show up in this list.

Using the pin listing on the next page, assign each programmable hardware pin to its corresponding physical hardware pin. To do this, click the *Port* dropdown menu (not the *Pin* dropdown!) and select the proper pin. Your physical pins are now attached to your *Top Design* circuit!

Pin Listings

Name	Port[Pin]	Name	Port[Pin]
LED B1	15[0]	SPST 1	2[3]
LED B2	15[1]	SPST 2	2[4]
LED B3	15[5]	SPST 3	2[5]
LED B4	0[0]	SPST 4	2[0]
LED B5	0[1]	SPST 5	12[5]
LED B6	0[5]	SPST 6	12[4]
LED B7	0[6]	SPST 7	12[3]
LED B8	0[7]	SPST 8	12[2]

Assigning a physical pin



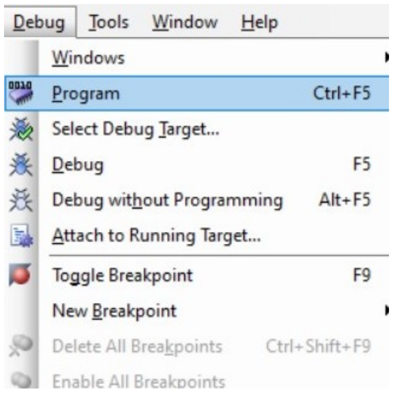
	Name	Port	Pin	Lock
🟩	LEDB_1	P15[0]	27	✓
🟩	LEDB_2	P15[1]	28	✓
🟩	LEDB_3	P15[5]	61	✓
	LEDB_4			

```
<auto-assign during build>
P0[0] OpAmp[2].vout
P0[1] OpAmp[0].vout
P0[2] OpAmp[0].vplus, SAR[1].ext_pin
P0[3] OpAmp[0].vminus, DSM[0].ext_pin_1
```

Programming the PSoC

The final step is to upload your work from the Creator project to your PSoC. Make sure the stick is plugged in via USB to your computer. Use the male USB connector labeled KITPROG. Then, in the top bar of Creator, click *Debug* > *Program*. If a message asks you to pick your debug target, select the PSoC you have plugged in. Wait for the project to compile (watch the *Output* area at the bottom of Creator to track progress). Once this is done, your changes will be on the PSoC. The programmed PSoC won't initially look that much different from before, but now when you flip a DIP switch to the ON position, the corresponding LED should toggle on!

For now, the PSoC is being powered from the USB cable you used to program it. In the future, if you disconnect the USB cable, you may power the PSoC using the breadboard power supply instead. However, **make sure to NEVER use both the power supply and the USB cable at the same time.**



Programming the PSoC

Wrapup

Think about what you have just accomplished. You have created a circuit essentially by just thinking about it. The ability to create a circuit using just software is practically a superpower. Circuits from previous labs were etched directly into circuit boards; if you wanted to change how they work, you would have to get a completely different board. On the PSoC, however, you can change circuits as quickly as you can click. For instance, you could quickly alter which DIP switches map to which LEDs on your board simply by redrawing the wires (or changing the physical pin mappings) and reprogramming the PSoC.

This lab only scratches the surface of what the PSoC is capable of. In the next lab, you will dig further into its bag of tricks, including its extensive component library.