# PSoC® Creator™
# Project Datasheet for Driver Station PWM Encoder Design

**Creation Time: 02/18/2016 16:54:39**
**User: Jim-THINK\Jim**
**Project: Driver Station PWM Encoder Design**
**Tool: PSoC Creator 3.3 CP1**

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone (USA): 800.858.1810
Phone (Intl): 408.943.2600
http://www.cypress.com

## Copyright

## Trademarks

## Philips I2C Patent Rights

Purchase of I2C components from Cypress or one of its sublicensed Associated Companies conveys a license under the Philips I2C Patent Rights to use these components in an I2C system, provided that the system conforms to the I2C Standard Specification as defined by Philips. As from October 1st, 2006 Philips Semiconductors has a new trade name, NXP Semiconductors.

## Disclaimer

CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. While reasonable precautions have been taken, Cypress assumes no responsibility for any errors that may appear in this document. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of a Cypress product in a life support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

## Flash Code Protection

Cypress products meet the specifications contained in their particular Cypress PSoC Datasheets. Cypress believes that its family of PSoC products is one of the most secure families of its kind on the market today, regardless of how they are used. There may be methods, unknown to Cypress, that can breach the code protection features. Any of these methods, to our knowledge, would be dishonest and possibly illegal. Neither Cypress nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as 'unbreakable.'

Cypress is willing to work with the customer who is concerned about the integrity of their code. Code protection is constantly evolving. We at Cypress are committed to continuously improving the code protection features of our products.

# Table of Contents

# 1 Overview

The Cypress PSoC 5 is a family of 32-bit devices with the following characteristics:

- High-performance 32-bit ARM Cortex-M3 core with a nested vectored interrupt controller (NVIC) and a high-performance DMA controller
- Digital system that includes configurable Universal Digital Blocks (UDBs) and specific function peripherals, such as USB, I2C and SPI
- Analog subsystem that includes 20-bit Delta Sigma converters (ADC), SAR ADCs, 8-bit DACs that can be configured for 12-bit operation, comparators, op amps and configurable switched capacitor (SC) and continuous time (CT) blocks to create PGAs, TIAs, mixers, and more
- Several types of memory elements, including SRAM, flash, and EEPROM
- Programming and debug system through JTAG, serial wire debug (SWD), and single wire viewer (SWV)
- Flexible routing to all pins

Figure 1 shows the major components of a typical CY8C58LP family member PSoC 5 device. For details on all the systems listed above, please refer to the PSoC 5 Technical Reference Manual .

Figure 1. CY8C58LP Device Family Block Diagram

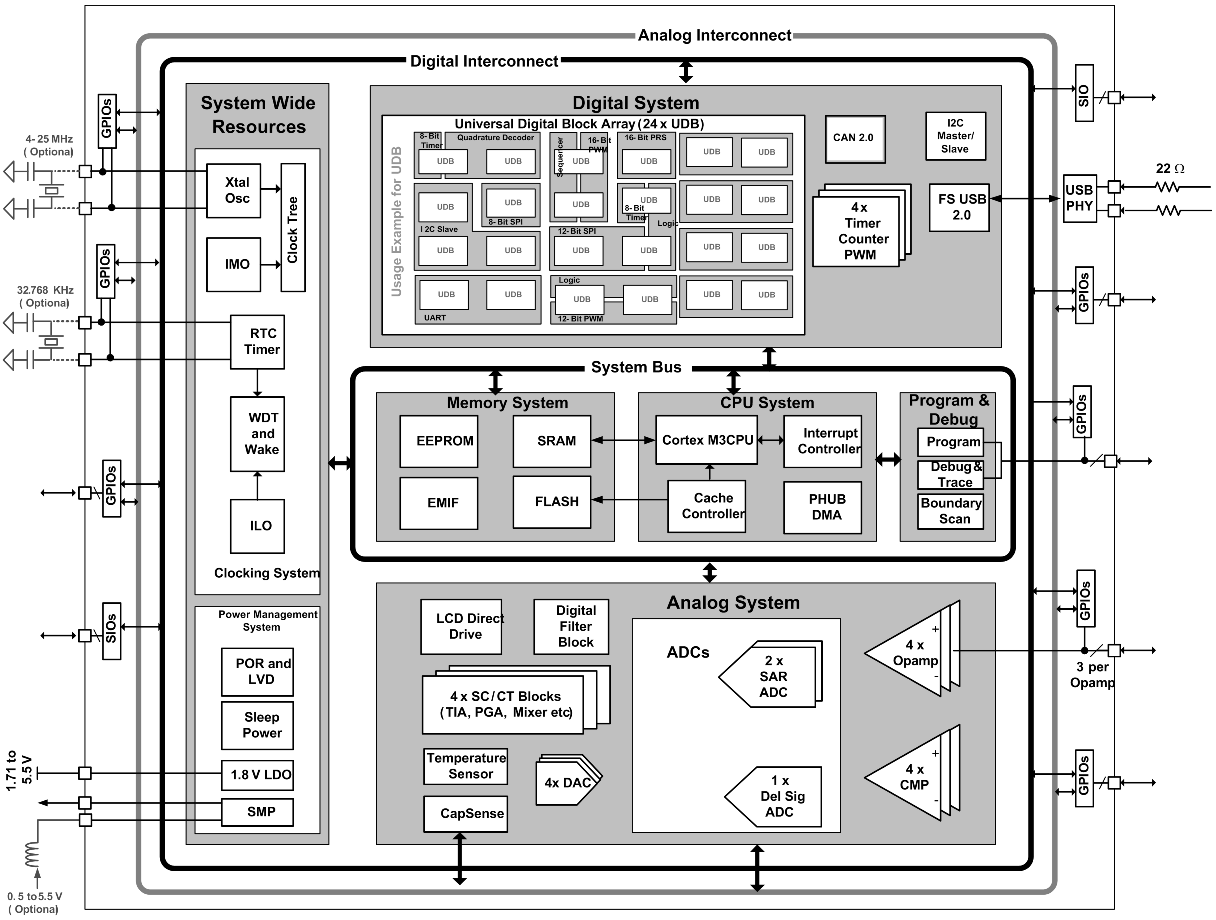


Table 1 lists the key characteristics of this device.

Table 1. Device Characteristics

| Name | Value |
|---|---|
| Part Number | CY8C5888LTI-LP097 |
| Package Name | 68-QFN |
| Architecture | PSoC 5 |
| Family | CY8C58LP |
| CPU speed (MHz) | 80 |
| Flash size (kBytes) | 256 |
| SRAM size (kBytes) | 64 |
| EEPROM size (Bytes) | 2048 |
| Vdd range (V) | 1.71 to 5.5 |
| Automotive qualified | No (Industrial Grade Only) |
| Temp range (Celcius) | -40 to 85 |
| JTAG ID | 0x2E161069 |

NOTE: The CPU speed noted above is the maximum available speed. The CPU is clocked by Bus Clock, listed in the System Clocks section below.

Table 2 lists the device resources that this design uses:

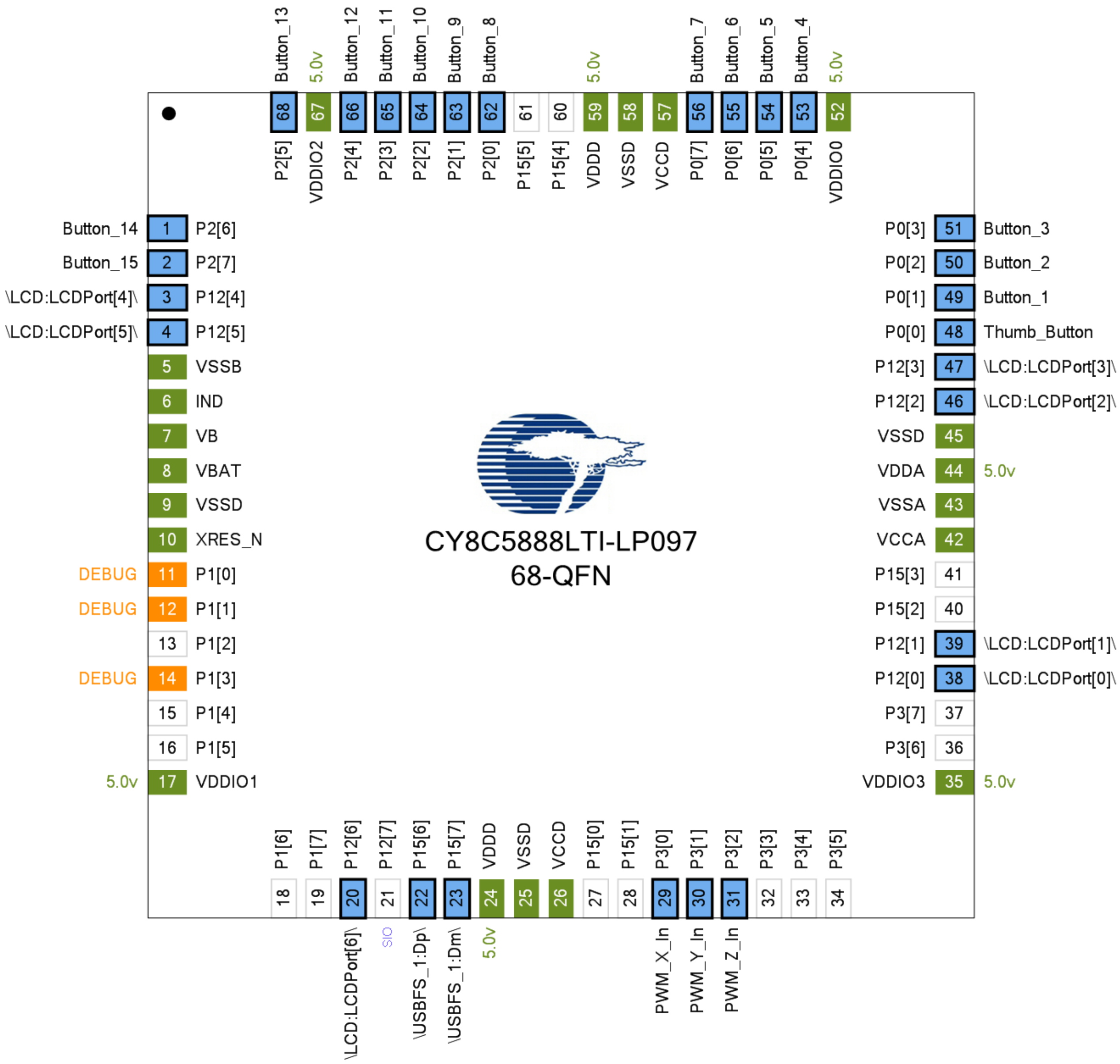
Table 2. Device Resources

| Resource Type | Used | Free | Max | % Used |
|---|---|---|---|---|
| Digital Clocks | 2 | 6 | 8 | 25.00 % |
| Analog Clocks | 0 | 4 | 4 | 0.00 % |
| CapSense Buffers | 0 | 2 | 2 | 0.00 % |
| Digital Filter Block | 0 | 1 | 1 | 0.00 % |
| Interrupts | 8 | 24 | 32 | 25.00 % |
| IO | 33 | 15 | 48 | 68.75 % |
| Segment LCD | 0 | 1 | 1 | 0.00 % |
| CAN 2.0b | 0 | 1 | 1 | 0.00 % |
| I2C | 0 | 1 | 1 | 0.00 % |
| USB | 1 | 0 | 1 | 100.00 % |
| DMA Channels | 0 | 24 | 24 | 0.00 % |
| Timer | 0 | 4 | 4 | 0.00 % |
| UDB | | | | |
| Macrocells | 6 | 186 | 192 | 3.13 % |
| Unique P-terms | 15 | 369 | 384 | 3.91 % |
| Total P-terms | 15 | | | |
| Datapath Cells | 2 | 22 | 24 | 8.33 % |
| Status Cells | 1 | 23 | 24 | 4.17 % |
| StatusI Registers | 1 | | | |
| Control Cells | 1 | 23 | 24 | 4.17 % |
| Control Registers | 1 | | | |
| Opamp | 0 | 4 | 4 | 0.00 % |
| Comparator | 0 | 4 | 4 | 0.00 % |
| Delta-Sigma ADC | 0 | 1 | 1 | 0.00 % |
| LPF | 0 | 2 | 2 | 0.00 % |
| SAR ADC | 0 | 2 | 2 | 0.00 % |
| Analog (SC/CT) Blocks | 0 | 4 | 4 | 0.00 % |
| DAC | | | | |
| VIDAC | 0 | 4 | 4 | 0.00 % |

# 2 Pins

Figure 2 shows the pin layout of this device.

Figure 2. Device Pin Layout



## 2.1 Hardware Pins

Table 3 contains information about the pins on this device in device pin order. (No connection ["n/c"] pins have been omitted.)

Table 3. Device Pins

| Pin | Port | Name | Type | Drive Mode | Reset State |
|---|---|---|---|---|---|
| 1 | P2[6] | Button_14 | Software Input | Res pull down | HiZ Analog Unb |
| 2 | P2[7] | Button_15 | Software Input | Res pull down | HiZ Analog Unb |
| 3 | P12[4] | \LCD:LCDPort[4]\ | Software Output | Strong drive | HiZ Analog Unb |
| 4 | P12[5] | \LCD:LCDPort[5]\ | Software Output | Strong drive | HiZ Analog Unb |
| 5 | VSSB | VSSB | Dedicated | | |
| 6 | IND | IND | Dedicated | | |
| 7 | VB | VB | Dedicated | | |
| 8 | VBAT | VBAT | Dedicated | | |
| 9 | VSSD | VSSD | Power | | |
| 10 | XRES_N | XRES_N | Dedicated | | |
| 11 | P1[0] | Debug:SWD_IO | Reserved | | |
| 12 | P1[1] | Debug:SWD_CK | Reserved | | |
| 13 | P1[2] | GPIO [unused] | | | HiZ Analog Unb |
| 14 | P1[3] | Debug:SWV | Reserved | | |
| 15 | P1[4] | GPIO [unused] | | | HiZ Analog Unb |
| 16 | P1[5] | GPIO [unused] | | | HiZ Analog Unb |
| 17 | VDDIO1 | VDDIO1 | Power | | |
| 18 | P1[6] | GPIO [unused] | | | HiZ Analog Unb |
| 19 | P1[7] | GPIO [unused] | | | HiZ Analog Unb |
| 20 | P12[6] | \LCD:LCDPort[6]\ | Software Output | Strong drive | HiZ Analog Unb |
| 21 | P12[7] | SIO [unused] | | | HiZ Analog Unb |
| 22 | P15[6] | USB:D+ | Reserved | | |
| 23 | P15[7] | USB:D- | Reserved | | |
| 24 | VDDD | VDDD | Power | | |
| 25 | VSSD | VSSD | Power | | |
| 26 | VCCD | VCCD | Power | | |
| 27 | P15[0] | GPIO [unused] | | | HiZ Analog Unb |
| 28 | P15[1] | GPIO [unused] | | | HiZ Analog Unb |
| 29 | P3[0] | PWM_X_In | Dgtl In | Res pull up/down | HiZ Analog Unb |
| 30 | P3[1] | PWM_Y_In | Software Input | Res pull up/down | HiZ Analog Unb |
| 31 | P3[2] | PWM_Z_In | Software Input | Res pull up/down | HiZ Analog Unb |
| 32 | P3[3] | GPIO [unused] | | | HiZ Analog Unb |
| 33 | P3[4] | GPIO [unused] | | | HiZ Analog Unb |
| 34 | P3[5] | GPIO [unused] | | | HiZ Analog Unb |
| 35 | VDDIO3 | VDDIO3 | Power | | |
| 36 | P3[6] | GPIO [unused] | | | HiZ Analog Unb |
| 37 | P3[7] | GPIO [unused] | | | HiZ Analog Unb |

| Pin | Port | Name | Type | Drive Mode | Reset State |
|---|---|---|---|---|---|
| 38 | P12[0] | \LCD:LCDPort[0]\ | Software Output | Strong drive | HiZ Analog Unb |
| 39 | P12[1] | \LCD:LCDPort[1]\ | Software Output | Strong drive | HiZ Analog Unb |
| 40 | P15[2] | GPIO [unused] | | | HiZ Analog Unb |
| 41 | P15[3] | GPIO [unused] | | | HiZ Analog Unb |
| 42 | VCCA | VCCA | Power | | |
| 43 | VSSA | VSSA | Power | | |
| 44 | VDDA | VDDA | Power | | |
| 45 | VSSD | VSSD | Power | | |
| 46 | P12[2] | \LCD:LCDPort[2]\ | Software Output | Strong drive | HiZ Analog Unb |
| 47 | P12[3] | \LCD:LCDPort[3]\ | Software Output | Strong drive | HiZ Analog Unb |
| 48 | P0[0] | Thumb_Button | Software Input | Res pull down | HiZ Analog Unb |
| 49 | P0[1] | Button_1 | Software Input | Res pull down | HiZ Analog Unb |
| 50 | P0[2] | Button_2 | Software Input | Res pull down | HiZ Analog Unb |
| 51 | P0[3] | Button_3 | Software Input | Res pull down | HiZ Analog Unb |
| 52 | VDDIO0 | VDDIO0 | Power | | |
| 53 | P0[4] | Button_4 | Software Input | Res pull down | HiZ Analog Unb |
| 54 | P0[5] | Button_5 | Software Input | Res pull down | HiZ Analog Unb |
| 55 | P0[6] | Button_6 | Software Input | Res pull down | HiZ Analog Unb |
| 56 | P0[7] | Button_7 | Software Input | Res pull down | HiZ Analog Unb |
| 57 | VCCD | VCCD | Power | | |
| 58 | VSSD | VSSD | Power | | |
| 59 | VDDD | VDDD | Power | | |
| 60 | P15[4] | GPIO [unused] | | | HiZ Analog Unb |
| 61 | P15[5] | GPIO [unused] | | | HiZ Analog Unb |
| 62 | P2[0] | Button_8 | Software Input | Res pull down | HiZ Analog Unb |
| 63 | P2[1] | Button_9 | Software Input | Res pull down | HiZ Analog Unb |
| 64 | P2[2] | Button_10 | Software Input | Res pull down | HiZ Analog Unb |
| 65 | P2[3] | Button_11 | Software Input | Res pull down | HiZ Analog Unb |
| 66 | P2[4] | Button_12 | Software Input | Res pull down | HiZ Analog Unb |
| 67 | VDDIO2 | VDDIO2 | Power | | |
| 68 | P2[5] | Button_13 | Software Input | Res pull down | HiZ Analog Unb |

Abbreviations used in Table 3 have the following meanings:

- Res pull down = Resistive pull down
- HiZ Analog Unb = Hi-Z Analog Unbuffered
- Dgtl In = Digital Input
- Res pull up/down = Resistive pull up/down

## 2.2 Hardware Ports

Table 4 contains information about the pins on this device in device port order. (No connection ["n/c"], power and dedicated pins have been omitted.)

Table 4. Device Ports

| Port | Pin | Name | Type | Drive Mode | Reset State |
|---|---|---|---|---|---|
| P0[0] | 48 | Thumb_Button | Software Input | Res pull down | HiZ Analog Unb |
| P0[1] | 49 | Button_1 | Software Input | Res pull down | HiZ Analog Unb |
| P0[2] | 50 | Button_2 | Software Input | Res pull down | HiZ Analog Unb |
| P0[3] | 51 | Button_3 | Software Input | Res pull down | HiZ Analog Unb |
| P0[4] | 53 | Button_4 | Software Input | Res pull down | HiZ Analog Unb |
| P0[5] | 54 | Button_5 | Software Input | Res pull down | HiZ Analog Unb |
| P0[6] | 55 | Button_6 | Software Input | Res pull down | HiZ Analog Unb |
| P0[7] | 56 | Button_7 | Software Input | Res pull down | HiZ Analog Unb |
| P1[0] | 11 | Debug:SWD_IO | Reserved | | |
| P1[1] | 12 | Debug:SWD_CK | Reserved | | |
| P1[2] | 13 | GPIO [unused] | | | HiZ Analog Unb |
| P1[3] | 14 | Debug:SWV | Reserved | | |
| P1[4] | 15 | GPIO [unused] | | | HiZ Analog Unb |
| P1[5] | 16 | GPIO [unused] | | | HiZ Analog Unb |
| P1[6] | 18 | GPIO [unused] | | | HiZ Analog Unb |
| P1[7] | 19 | GPIO [unused] | | | HiZ Analog Unb |
| P12[0] | 38 | \LCD:LCDPort[0]\ | Software Output | Strong drive | HiZ Analog Unb |
| P12[1] | 39 | \LCD:LCDPort[1]\ | Software Output | Strong drive | HiZ Analog Unb |
| P12[2] | 46 | \LCD:LCDPort[2]\ | Software Output | Strong drive | HiZ Analog Unb |
| P12[3] | 47 | \LCD:LCDPort[3]\ | Software Output | Strong drive | HiZ Analog Unb |
| P12[4] | 3 | \LCD:LCDPort[4]\ | Software Output | Strong drive | HiZ Analog Unb |
| P12[5] | 4 | \LCD:LCDPort[5]\ | Software Output | Strong drive | HiZ Analog Unb |
| P12[6] | 20 | \LCD:LCDPort[6]\ | Software Output | Strong drive | HiZ Analog Unb |
| P12[7] | 21 | SIO [unused] | | | HiZ Analog Unb |
| P15[0] | 27 | GPIO [unused] | | | HiZ Analog Unb |
| P15[1] | 28 | GPIO [unused] | | | HiZ Analog Unb |
| P15[2] | 40 | GPIO [unused] | | | HiZ Analog Unb |
| P15[3] | 41 | GPIO [unused] | | | HiZ Analog Unb |
| P15[4] | 60 | GPIO [unused] | | | HiZ Analog Unb |
| P15[5] | 61 | GPIO [unused] | | | HiZ Analog Unb |
| P15[6] | 22 | USB:D+ | Reserved | | |
| P15[7] | 23 | USB:D- | Reserved | | |

| Port | Pin | Name | Type | Drive Mode | Reset State |
|---|---|---|---|---|---|
| P2[0] | 62 | Button_8 | Software Input | Res pull down | HiZ Analog Unb |
| P2[1] | 63 | Button_9 | Software Input | Res pull down | HiZ Analog Unb |
| P2[2] | 64 | Button_10 | Software Input | Res pull down | HiZ Analog Unb |
| P2[3] | 65 | Button_11 | Software Input | Res pull down | HiZ Analog Unb |
| P2[4] | 66 | Button_12 | Software Input | Res pull down | HiZ Analog Unb |
| P2[5] | 68 | Button_13 | Software Input | Res pull down | HiZ Analog Unb |
| P2[6] | 1 | Button_14 | Software Input | Res pull down | HiZ Analog Unb |
| P2[7] | 2 | Button_15 | Software Input | Res pull down | HiZ Analog Unb |
| P3[0] | 29 | PWM_X_In | Dgtl In | Res pull up/down | HiZ Analog Unb |
| P3[1] | 30 | PWM_Y_In | Software Input | Res pull up/down | HiZ Analog Unb |
| P3[2] | 31 | PWM_Z_In | Software Input | Res pull up/down | HiZ Analog Unb |
| P3[3] | 32 | GPIO [unused] | | | HiZ Analog Unb |
| P3[4] | 33 | GPIO [unused] | | | HiZ Analog Unb |
| P3[5] | 34 | GPIO [unused] | | | HiZ Analog Unb |
| P3[6] | 36 | GPIO [unused] | | | HiZ Analog Unb |
| P3[7] | 37 | GPIO [unused] | | | HiZ Analog Unb |

Abbreviations used in Table 4 have the following meanings:

- Res pull down = Resistive pull down
- HiZ Analog Unb = Hi-Z Analog Unbuffered
- Dgtl In = Digital Input
- Res pull up/down = Resistive pull up/down

## 2.3 Software Pins

Table 5 contains information about the software pins on this device in alphabetical order. (Only software-accessible pins are shown.)

Table 5. Software Pins

| Name | Port | Type | Reset State |
|---|---|---|---|
| \LCD:LCDPort[0]\ | P12[0] | Software Output | HiZ Analog Unb |
| \LCD:LCDPort[1]\ | P12[1] | Software Output | HiZ Analog Unb |
| \LCD:LCDPort[2]\ | P12[2] | Software Output | HiZ Analog Unb |
| \LCD:LCDPort[3]\ | P12[3] | Software Output | HiZ Analog Unb |
| \LCD:LCDPort[4]\ | P12[4] | Software Output | HiZ Analog Unb |
| \LCD:LCDPort[5]\ | P12[5] | Software Output | HiZ Analog Unb |
| \LCD:LCDPort[6]\ | P12[6] | Software Output | HiZ Analog Unb |
| Button_1 | P0[1] | Software Input | HiZ Analog Unb |
| Button_10 | P2[2] | Software Input | HiZ Analog Unb |
| Button_11 | P2[3] | Software Input | HiZ Analog Unb |
| Button_12 | P2[4] | Software Input | HiZ Analog Unb |
| Button_13 | P2[5] | Software Input | HiZ Analog Unb |
| Button_14 | P2[6] | Software Input | HiZ Analog Unb |
| Button_15 | P2[7] | Software Input | HiZ Analog Unb |
| Button_2 | P0[2] | Software Input | HiZ Analog Unb |
| Button_3 | P0[3] | Software Input | HiZ Analog Unb |
| Button_4 | P0[4] | Software Input | HiZ Analog Unb |
| Button_5 | P0[5] | Software Input | HiZ Analog Unb |
| Button_6 | P0[6] | Software Input | HiZ Analog Unb |
| Button_7 | P0[7] | Software Input | HiZ Analog Unb |
| Button_8 | P2[0] | Software Input | HiZ Analog Unb |
| Button_9 | P2[1] | Software Input | HiZ Analog Unb |
| Debug:SWD_CK | P1[1] | Reserved | |
| Debug:SWD_IO | P1[0] | Reserved | |
| Debug:SWV | P1[3] | Reserved | |
| PWM_X_In | P3[0] | Dgtl In | HiZ Analog Unb |

| Name | Port | Type | Reset State |
|---|---|---|---|
| PWM_Y_In | P3[1] | Software Input | HiZ Analog Unb |
| PWM_Z_In | P3[2] | Software Input | HiZ Analog Unb |
| Thumb_Button | P0[0] | Software Input | HiZ Analog Unb |
| USB:D- | P15[7] | Reserved | |
| USB:D+ | P15[6] | Reserved | |

Abbreviations used in Table 5 have the following meanings:

- HiZ Analog Unb = Hi-Z Analog Unbuffered
- Dgtl In = Digital Input

For more information on reading, writing and configuring pins, please refer to:

- Pins chapter in the System Reference Guide
  - CyPins API routines
- Programming Application Interface section in the cy_pins component datasheet

# 3 System Settings

## 3.1 System Configuration

Table 6. System Configuration Settings

| Name | Value |
|---|---|
| Device Configuration Mode | Uncompressed |
| Enable Error Correcting Code (ECC) | False |
| Store Configuration Data in ECC Memory | True |
| Instruction Cache Enabled | True |
| Enable Fast IMO During Startup | False |
| Unused Bonded IO | Allow but warn |
| Heap Size (bytes) | 0x80 |
| Stack Size (bytes) | 0x0800 |
| Include CMSIS Core Peripheral Library Files | True |

## 3.2 System Debug Settings

Table 7. System Debug Settings

| Name | Value |
|---|---|
| Debug Select | SWD+SWV (serial wire debug and viewer) |
| Enable Device Protection | False |
| Embedded Trace (ETM) | False |
| Use Optional XRES | False |

## 3.3 System Operating Conditions

Table 8. System Operating Conditions

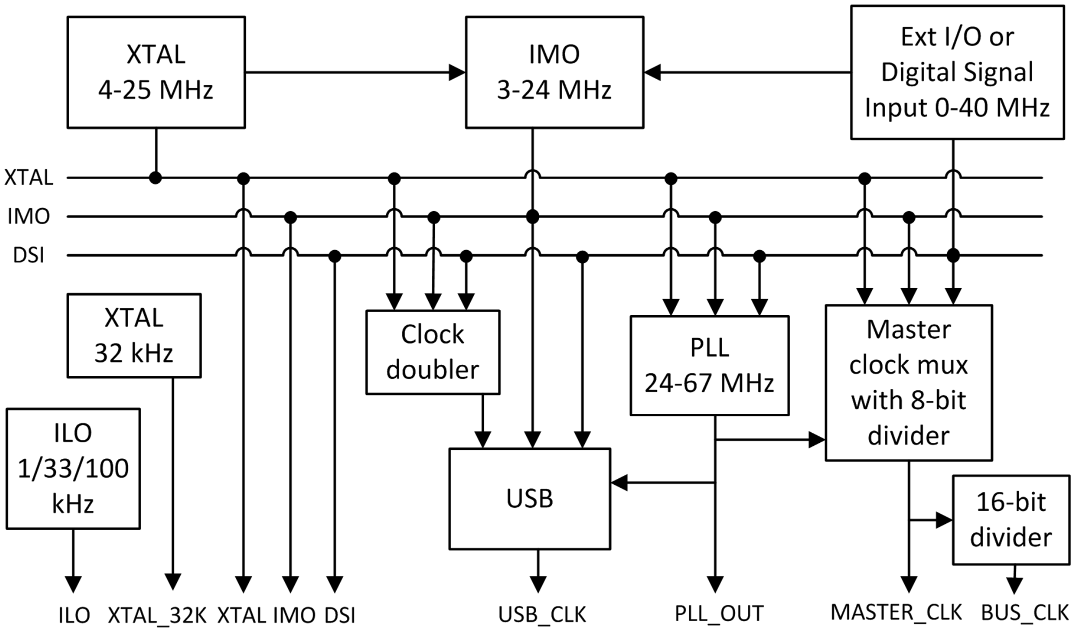
| Name | Value |
|---|---|
| Variable VDDA | False |
| VDDA (V) | 5.0 |
| VDDD (V) | 5.0 |
| VDDIO0 (V) | 5.0 |
| VDDIO1 (V) | 5.0 |
| VDDIO2 (V) | 5.0 |
| VDDIO3 (V) | 5.0 |
| Temperature Range | -40C - 85/125C |

# 4 Clocks

The clock system includes these clock resources:

- Four internal clock sources increase system integration:
  - 3 to 74.7 MHz Internal Main Oscillator (IMO) ±1% at 3 MHz
  - 1 kHz, 33 kHz, and 100 kHz Internal Low Speed Oscillator (ILO) outputs
  - 12 to 80 MHz clock doubler output, sourced from IMO, MHz External Crystal Oscillator (MHzECO), and Digital System Interconnect (DSI)
  - 24 to 80 MHz fractional Phase-Locked Loop (PLL) sourced from IMO, MHzECO, and DSI
- Clock generated using a DSI signal from an external I/O pin or other logic
- Two external clock sources provide high precision clocks:
  - 4 to 25 MHz External Crystal Oscillator (MHzECO)
  - 32.768 kHz External Crystal Oscillator (kHzECO) for Real Time Clock (RTC)
- Dedicated 16-bit divider for bus clock
- Eight individually sourced 16-bit clock dividers for the digital system peripherals
- Four individually sourced 16-bit clock dividers with skew for the analog system peripherals
- IMO has a USB mode that synchronizes to USB host traffic, requiring no external crystal for USB. (USB equipped parts only)

Figure 3. System Clock Configuration



## 4.1 System Clocks

Table 9 lists the system clocks used in this design.

Table 9. System Clocks

| Name | Domain | Source | Desired Freq | Nominal Freq | Accuracy (%) | Start at Reset | Enabled |
|---|---|---|---|---|---|---|---|
| BUS_CLK | DIGITAL | MASTER_CLK | ? MHz | 48 MHz | ±0.25 | True | True |
| MASTER_CLK | DIGITAL | PLL_OUT | ? MHz | 48 MHz | ±0.25 | True | True |
| USB_CLK | DIGITAL | IMO | 48 MHz | 48 MHz | ±0.25 | False | True |
| PLL_OUT | DIGITAL | IMO | 48 MHz | 48 MHz | ±0.25 | True | True |
| IMO | DIGITAL |  | 24 MHz | 24 MHz | ±0.25 | True | True |
| ILO | DIGITAL |  | ? MHz | 100 kHz | -55,+100 | True | True |
| XTAL 32kHz | DIGITAL |  | 32.768 kHz | ? MHz | ±0 | False | False |
| Digital Signal | DIGITAL |  | ? MHz | ? MHz | ±0 | False | False |
| XTAL | DIGITAL |  | 25 MHz | ? MHz | ±0 | False | False |

## 4.2 Local and Design Wide Clocks

Local clocks drive individual analog and digital blocks. Design wide clocks are a user-defined optimization, where two or more analog or digital blocks that share a common clock profile (frequency, etc) can be driven from the same clock divider output source.

Figure 4. Local and Design Wide Clock Configuration

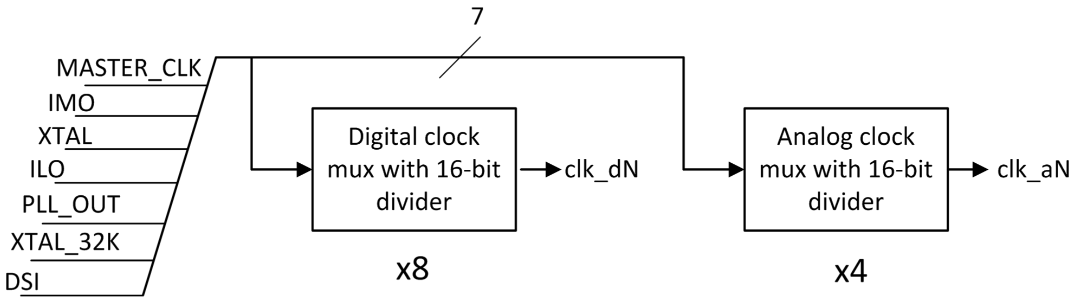

7
MASTER_CLK
IMO
XTAL
ILO
PLL_OUT
XTAL_32K
DSI
Digital clock mux with 16-bit divider
clk_dN
x8
Analog clock mux with 16-bit divider
clk_aN
x4

Table 10 lists the design wide clocks used in this design.

Table 10. Design Wide Clocks

| Name | Domain | Source | Desired Freq | Nominal Freq | Accuracy (%) | Start at Reset | Enabled |
|---|---|---|---|---|---|---|---|
| dwClock_1 | DIGITAL | MASTER_CLK | 24 MHz | 24 MHz | ±0.25 | True | True |

Table 11 lists the local clocks used in this design.

Table 11. Local Clocks

| Name | Domain | Source | Desired Freq | Nominal Freq | Accuracy (%) | Start at Reset | Enabled |
|---|---|---|---|---|---|---|---|
| USBFS_1_-Clock_vbus | DIGITAL | BUS_CLK | ? MHz | 48 MHz | ±0.25 | True | True |
| CLK_1MHZ | DIGITAL | MASTER_CLK | 1 MHz | 1 MHz | ±0.25 | True | True |

For more information on clocking resources, please refer to:

- Clocking System chapter in the PSoC 5 Technical Reference Manual
- Clocking chapter in the System Reference Guide
  - CyPLL API routines
  - CyIMO API routines
  - CyILO API routines
  - CyMaster API routines
  - CyXTAL API routines

# 5 Interrupts and DMAs

## 5.1 Interrupts

This design contains the following interrupt components: (0 is the highest priority)

Table 12. Interrupts

| Name | Priority | Vector |
|---|---|---|
| isr_Capture | 7 | 1 |
| USBFS_1_arb_int | 7 | 22 |
| USBFS_1_bus_reset | 7 | 23 |
| USBFS_1_dp_int | 7 | 12 |
| USBFS_1_ep_0 | 7 | 24 |
| USBFS_1_ep_1 | 7 | 0 |
| USBFS_1_ord_int | 7 | 25 |
| USBFS_1_sof_int | 7 | 21 |

For more information on interrupts, please refer to:

- Interrupt Controller chapter in the PSoC 5 Technical Reference Manual
- Interrupts chapter in the System Reference Guide
  - CyInt API routines and related registers
- Datasheet for cy_isr component

## 5.2 DMAs

This design contains no DMA components.

# 6 Flash Memory

PSoC 5 devices offer a host of Flash protection options and device security features that you can leverage to meet the security and protection requirements of an application. These requirements range from protecting configuration settings or Flash data to locking the entire device from external access.

Table 13 lists the Flash protection settings for your design.

Table 13. Flash Protection Settings

| Start Address | End Address | Protection Level |
|---|---|---|
| 0x0 | 0x3FFFF | U - Unprotected |

Flash memory is organized as rows with each row of flash having 256 bytes. Each flash row can be assigned one of four protection levels:

- U - Unprotected
- F - Factory Upgrade
- R - Field Upgrade
- W - Full Protection

For more information on Flash memory and protection, please refer to:

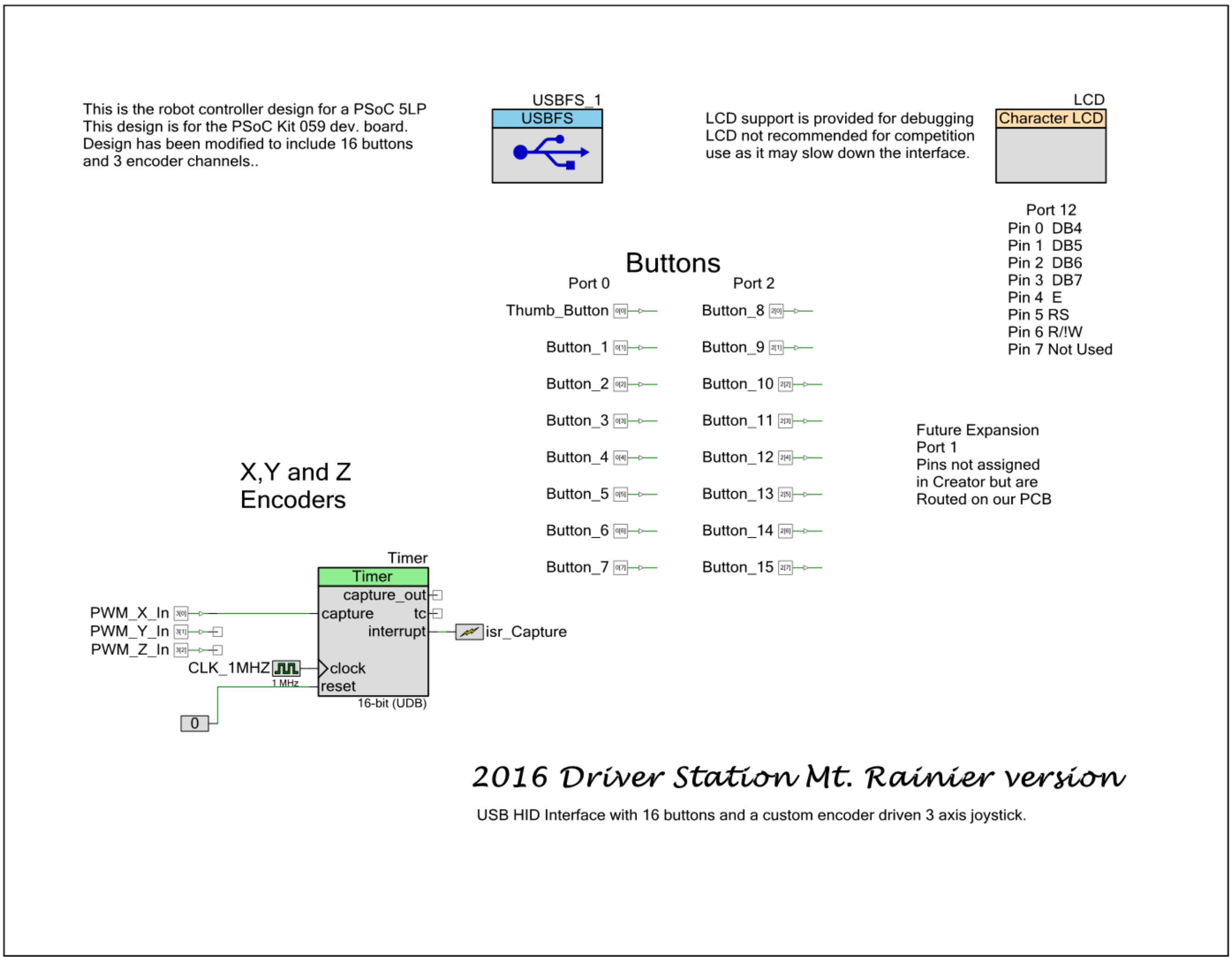
- Flash Protection chapter in the PSoC 5 Technical Reference Manual
- Flash and EEPROM chapter in the System Reference Guide
  - CyWrite API routines
  - CyFlash API routines

# 7 Design Contents

This design's schematic content consists of the following 6 schematic sheets:

## 7.1 Schematic Sheet: Top Design

Figure 5. Schematic Sheet: Top Design



This is the robot controller design for a PSoC 5LP
This design is for the PSoC Kit 059 dev. board.
Design has been modified to include 16 buttons
and 3 encoder channels..

LCD support is provided for debugging
LCD not recommended for competition
use as it may slow down the interface.

Future Expansion
Port 1
Pins not assigned
in Creator but are
Routed on our PCB

2016 Driver Station Mt. Rainier version

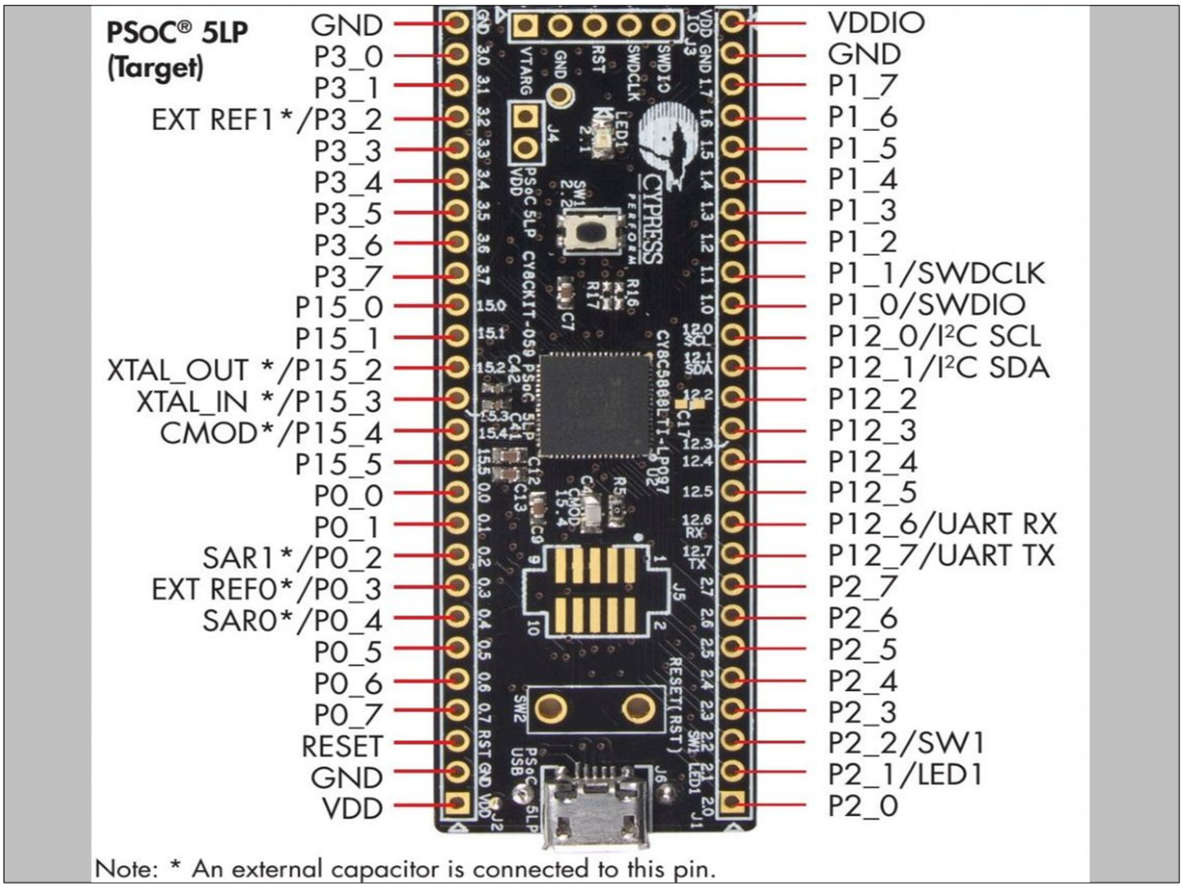
USB HID Interface with 16 buttons and a custom encoder driven 3 axis joystick.

This schematic sheet contains the following component instances:

- Instance LCD (type: CharLCD_v2_10)
- Instance Timer (type: Timer_v2_70)
- Instance USBFS_1 (type: USBFS_v2_80)

## 7.2 Schematic Sheet: Board Graphic

Figure 6. Schematic Sheet: Board Graphic



Note: * An external capacitor is connected to this pin.

## 7.3 Schematic Sheet: LCD Notes

Figure 7. Schematic Sheet: LCD Notes

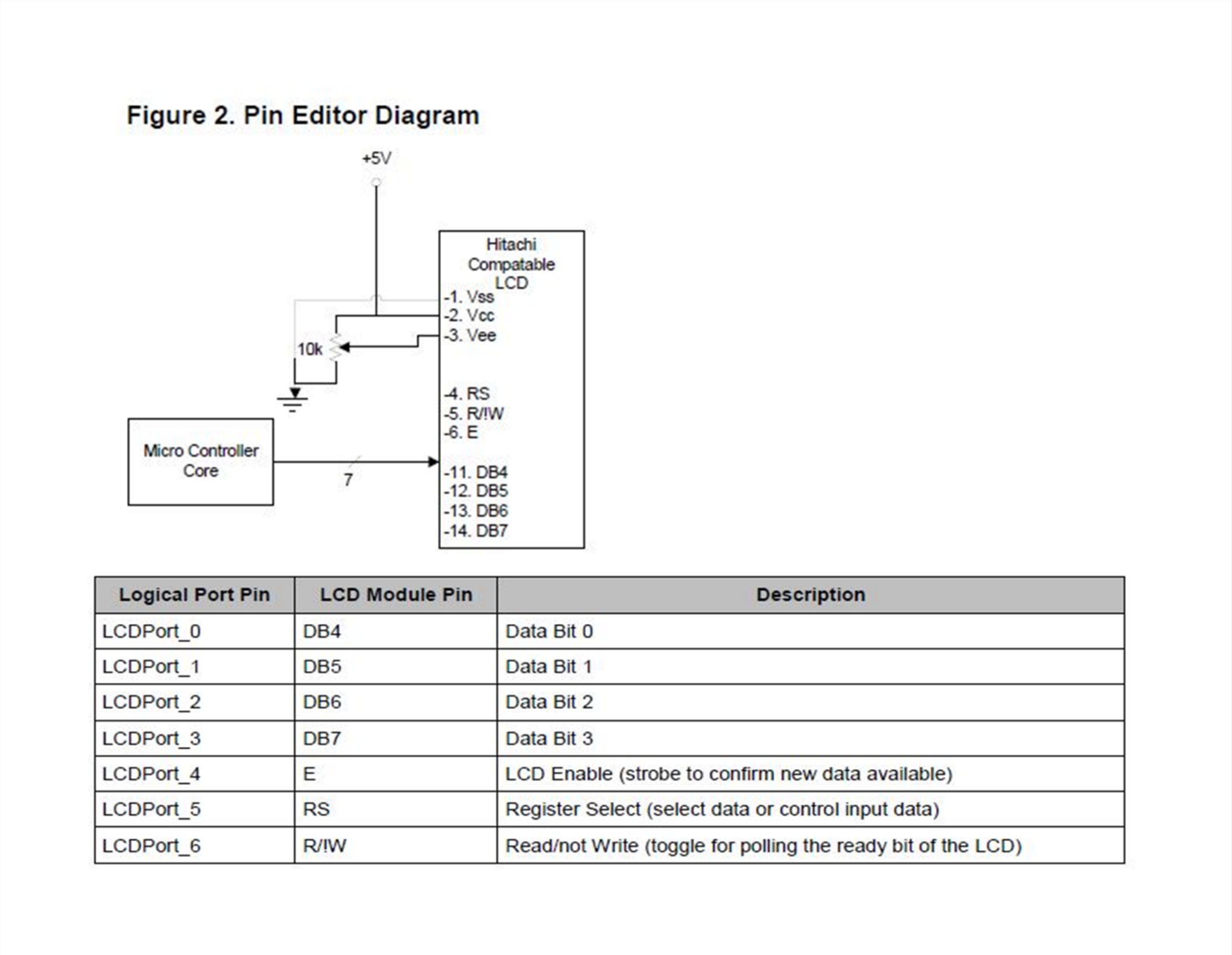
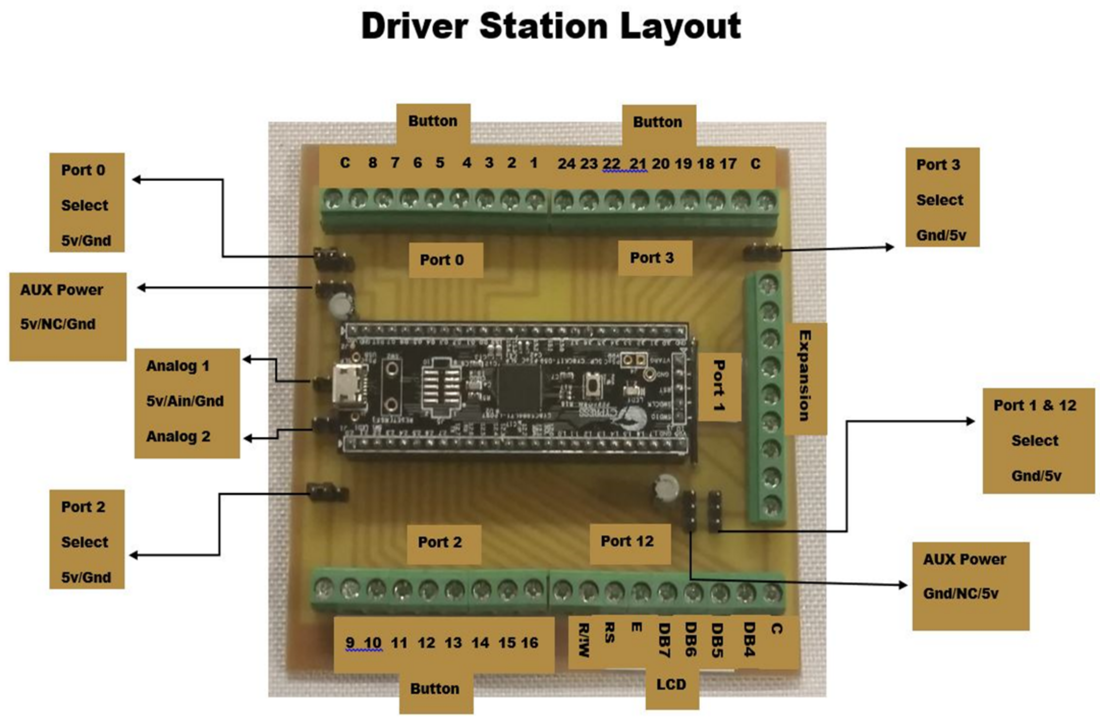


Figure 2. Pin Editor Diagram

| Logical Port Pin | LCD Module Pin | Description |
|---|---|---|
| LCDPort_0 | DB4 | Data Bit 0 |
| LCDPort_1 | DB5 | Data Bit 1 |
| LCDPort_2 | DB6 | Data Bit 2 |
| LCDPort_3 | DB7 | Data Bit 3 |
| LCDPort_4 | E | LCD Enable (strobe to confirm new data available) |
| LCDPort_5 | RS | Register Select (select data or control input data) |
| LCDPort_6 | R/!W | Read/not Write (toggle for polling the ready bit of the LCD) |

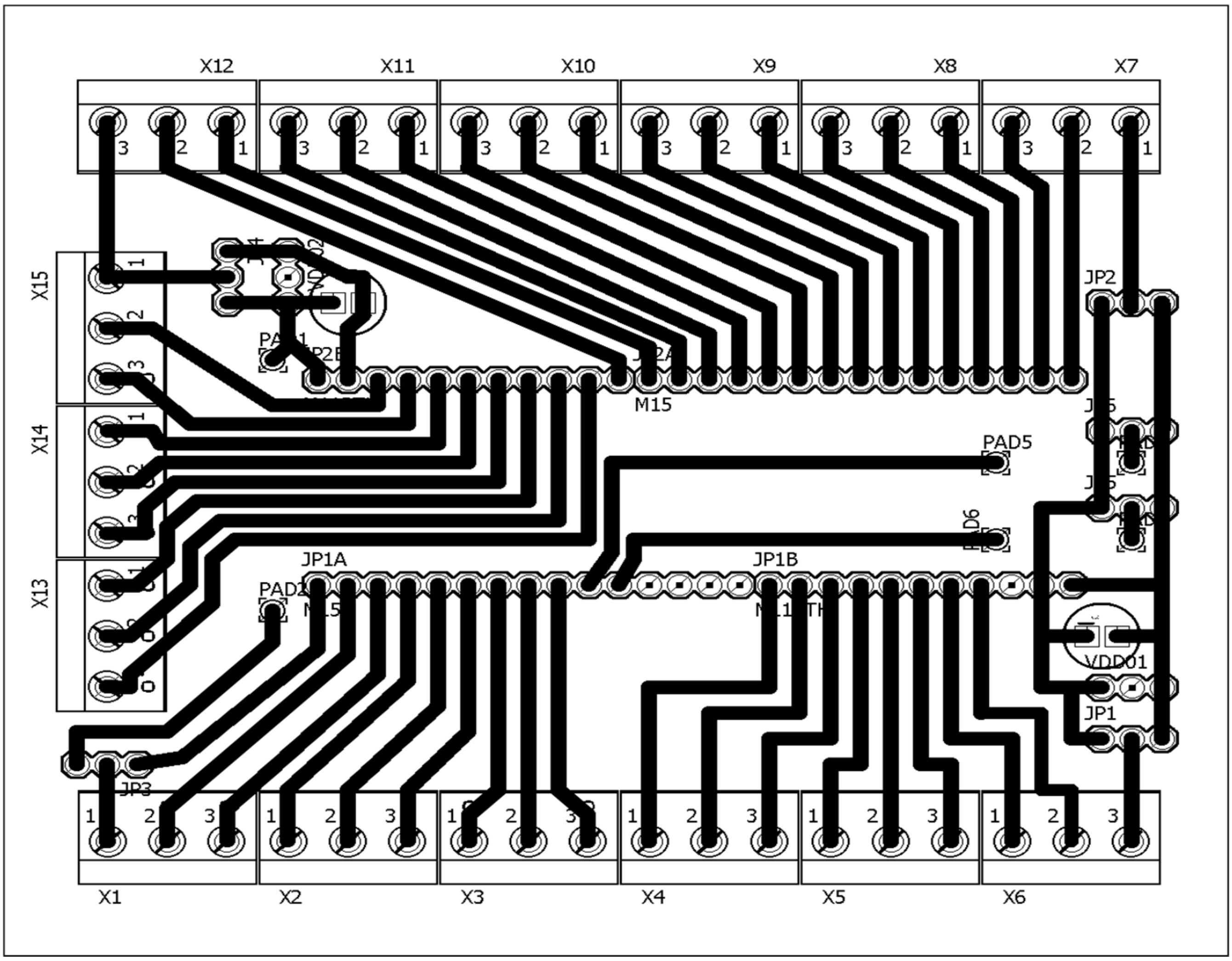
## 7.4 Schematic Sheet: Wiring Layout

Figure 8. Schematic Sheet: Wiring Layout



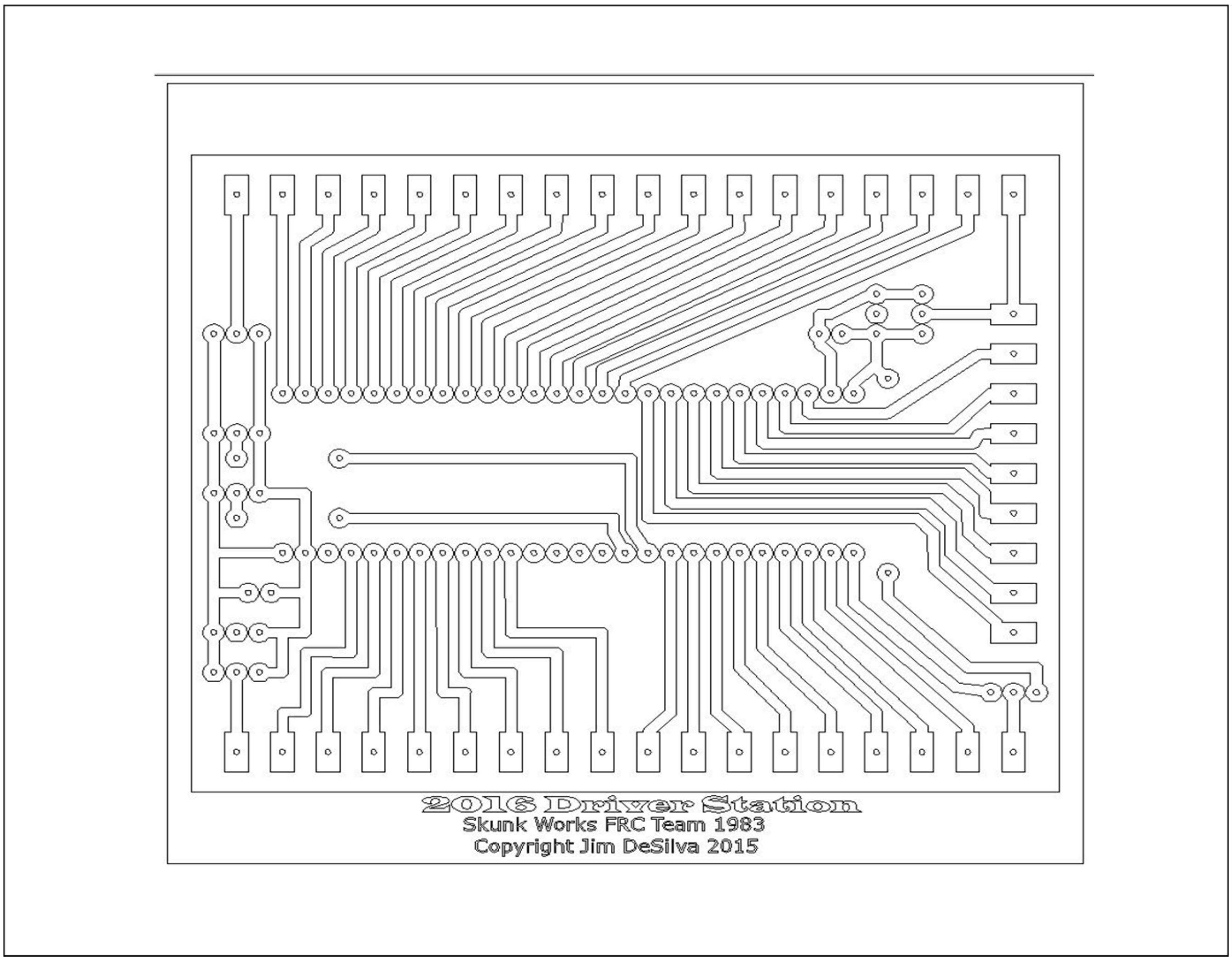
## 7.5 Schematic Sheet: PCB Art 1

Figure 9. Schematic Sheet: PCB Art 1



## 7.6 Schematic Sheet: PCB Bottom View

Figure 10. Schematic Sheet: PCB Bottom View



# 8 Components

## 8.1 Component type: CharLCD [v2.10]

### *8.1.1 Instance LCD*

**Description: Character LCD Component**
**Instance type: CharLCD [v2.10]**
**Datasheet: online component datasheet for CharLCD**

Table 14. Component Parameters for LCD

| Parameter Name | Value | Description |
|---|---|---|
| ConversionRoutines | true | Defines if the conversion routines will be included in the project. |
| CustomCharacterSet | None | Defines the type of custom character set (User defined, Vertical or Horizontal bargraph). Based on the selection a look-up table with proper characters representation will be generated in the source code. |

## 8.2 Component type: Timer [v2.70]

### *8.2.1 Instance Timer*

**Description: 8, 16, 24 or 32-bit Timer**
**Instance type: Timer [v2.70]**
**Datasheet: online component datasheet for Timer**

Table 15. Component Parameters for Timer

| Parameter Name | Value | Description |
|---|---|---|
| CaptureAlternatingFall | false | Enables data capture on either edge but not until a valid falling edge is detected first. |
| CaptureAlternatingRise | false | Enables data capture on either edge but not until a valid rising edge is detected first. |
| CaptureCount | 2 | The CaptureCount parameter works as a divider on the hardware input "capture". A CaptureCount value of 2 would result in an actual capture taking place every other time the input "capture" is changed. |
| CaptureCounterEnabled | false | Enables the capture counter to count capture events (up to 127) before a capture is triggered. |
| CaptureMode | Software Controlled | This parameter defines the capture input signal requirements to trigger a valid capture event |

| Parameter Name | Value | Description |
|---|---|---|
| EnableMode | Software Only | This parameter specifies the methods in enabling the component. Hardware mode makes the enable input pin visible. Software mode may reduce the resource usage if not enabled. |
| FixedFunction | false | Configures the component to use fixed function HW block instead of the UDB implementation. |
| InterruptOnCapture | true | Parameter to check whether interrupt on a capture event is enabled or disabled. |
| InterruptOnFIFOFull | false | Parameter to check whether interrupt on a FIFO Full event is enabled disabled. |
| InterruptOnTC | false | Parameter to check whether interrupt on a TC is enabled or disabled. |
| NumberOfCaptures | 1 | Number of captures allowed until the counter is cleared or disabled. |
| Period | 65535 | Defines the timer period (This is also the reload value when terminal count is reached) |
| Resolution | 16 | Defines the resolution of the hardware. This parameter affects how many bits are used in the Period counter and defines the maximum resolution of the internal component signals. |
| RunMode | Continuous | Defines the hardware to run continuously, run until a terminal count is reached or run until an interrupt event is triggered. |
| TriggerMode | None | Defines the required trigger input signal to cause a valid trigger enable of the timer |

## 8.3 Component type: USBFS [v2.80]

### *8.3.1 Instance USBFS_1*

**Description: USB 2.0 Full Speed Device Framework**
**Instance type: USBFS [v2.80]**
**Datasheet: online component datasheet for USBFS**

Table 16. Component Parameters for USBFS_1

| Parameter Name | Value | Description |
|---|---|---|
| EnableCDCApi | false | Enables additional high level API's that allow the CDC device to be used similar to a UART device. |
| EnableMidiApi | true | Enables additional high level MIDI API's. |

| Parameter Name | Value | Description |
|---|---|---|
| endpointMA | MA_Static | Endpoint memory allocation |
| endpointMM | EP_Manual | Endpoint memory management |
| epDMAautoOptimization | false | This parameter enables resource optimization for DMA with Automatic Memory Management mode. Set this parameter value to true only when a single IN endpoint is present in the device. Enabling this parameter in a multi IN endpoint device configuration causes undesired effects. |
| extern_cls | false | This parameter allows for user or other component to implement his own handler for Class requests. USBFS_-DispatchClassRqst() function should be implemented if this parameter enabled. |
| extern_vbus | false | This parameter enables external VBUSDET input. |
| extern_vnd | false | This parameter allows for user or other component to implement his own handler for Vendor specific requests. USBFS_HandleVendorRqst() function should be implemented if this parameter enabled. |
| extJackCount | 0 | Max number of External MIDI IN Jack or OUT Jack descriptors |
| max_interfaces_num | 1 | Defines maximum interfaces number |
| Mode | false | Specifies whether the implementation will create API for interfacing to UART component(s) for a corresponding set of external MIDI connections. |
| mon_vbus | false | The mon_vbus parameter adds a single VBUS monitor pin to the design. This pin must be connected to VBUS and must be assigned in the pin editor. |
| out_sof | false | The out_sof parameter enables Start-of-Frame output. |
| Pid | F232 | Product ID |
| Vid | 04B4 | Vendor ID |

# 9 Other Resources

The following documents contain important information on Cypress software APIs that might be relevant to this design:

- Standard Types and Defines chapter in the System Reference Guide
    - Software base types
    - Hardware register types
    - Compiler defines
    - Cypress API return codes
    - Interrupt types and macros
- Registers
    - The full PSoC 5 register map is covered in the PSoC 5 Registers Technical Reference Manual
    - Register Access chapter in the System Reference Guide
        - CY_GET API routines
        - CY_SET API routines
- System Functions chapter in the System Reference Guide
    - General API routines
    - CyDelay API routines
    - CyVd Voltage Detect API routines
- Power Management
    - Power Supply and Monitoring chapter in the PSoC 5 Technical Reference Manual
    - Low Power Modes chapter in the PSoC 5 Technical Reference Manual
    - Power Management chapter in the System Reference Guide
        - CyPm API routines
- Watchdog Timer chapter in the System Reference Guide
    - CyWdt API routines
- Cache Management
    - Cache Controller chapter in the PSoC 5 Technical Reference Manual
    - Cache chapter in the System Reference Guide
        - CyFlushCache() API routine