Lab n° 0
Real-Time systems [ELEC-H-410]
Introduction to PSoC microcontrollers and logic analyzers

2020–2021

Organisation

The course ELEC-H-410 “Real-Time Systems” has 9 practical work sessions divided into:

- 1 introduction lab to the PSoC microcontroller and logic analyzers;
- 2 labs about a real-time OS: FreeRTOS ;
- 4 labs to realize a project using the concepts previously acquired;
- 2 labs on the CAN bus communications, including an escape game.

You will be using a PSoC 5LP Development kit throughout the course.

Useful documentation:

- Getting started with PSoC 5LP: `https://www.cypress.com/file/41436/download`
- Video example on how to use the PSoC: `https://www.cypress.com/video-library/PSoC`
- The extension board schematics: `Extension_PSoC.pdf`
- Getting started with the Logic Analyzer: `https://learn.sparkfun.com/tutorials/using-the-usb-logic-analyzer-with-sigrok-pulseview`

Lab kit contents

Your lab kit should contain the following items:

- A PSoC microcontroller mounted on an extension board, with:
  - a 12-character keyboard;
  - a male-USB to female-USB cable;
- a logic analyzer, with:
  - an USB-A to USB-B-Mini cable;
  - ten female-to-female cables;
  - ten male-to-male pin headers.

If your lab kit is missing something, contact us as soon as possible.

1 Introduction to the PSoC microcontroller

The first part of the lab is to help you familiarize yourself with the PSoC microcontroller. For students who have had the course ELEC-H-310, this will mainly be a reminder of the course's labs.

You should start by installing the PSoC Creator IDE, as explained in Appendix A. If you are not confident with C programming, read `C_language_for_uC.pdf`.

In the following subsections, you will learn how to instantiate and use some basic hardware components of the PSoC microcontroller:

– GPIOs (including LEDs and pushbuttons of the extension board);
– the LCD screen;
– the keyboard.

You can always refer to refer to the PSoC tutorials (`www.cypress.com/psoc101`) or the datasheet of the hardware elements (these can be found by double-clicking on a component in the `TopDesign.cysch` file, and clicking on the button `Datasheet`).

1.1 General purpose input-output (GPIO)

GPIOs are the most basic hardware elements, allowing to provide input commands to the microcontroller and receive output signals from the microcontroller. You can find tutorials of how to use GPIOs on the following website:

– `www.cypress.com/psoc101`, Lesson 1: "1. Software Output Pins"
– `www.cypress.com/psoc101`, Lesson 2: "2. Software Input Pins"

The following exercise will help you to interface two sort of IOs : push-buttons and LEDs. The following (partial) list of functions are useful for using the GPIOs:

– `gpioName_Read()`: Read a GPIO value;
– `gpioName_Write(value)`: Write a binary value to a GPIO;
– `CyDelay()`: use this function to put the microcontroller to sleep for a certain amount of milliseconds between two iterations of the for loop.

Exercise 1. Open the `Lab0` project. In this project, the different hardware components of the extension board have already been instantiated.

– Identify which hardware elements have been instantiated in the `TopDesign.cysch` file. You can have the details of a hardware component by double-clicking on it.
– In the `Lab0.cydwr` file, the instantiated hardware elements have been associated with the pins of the PSoC microcontroller. Can you match the pin numbers of the PSoC with the hardware elements of the extension board by using the electrical schematics in Appendix B?
– Now write the code in the `main.c` file such that the LEDs D1-D4 turns on when the corresponding button SW1-SW4 is pushed, and turns off when this button is released.
– Build, compile and load it on the PSoC board, then check its behavior (the shortcut buttons to build, compile and program the PSoC are located on the ribbon on the top left of the PSoC IDE).

It is important to note that, in general, microcontroller are not designed to deliver high currents. If we were using high-brightness LEDs, we would need to use *buffer* circuits (such as the `74ACT244`) to provide larger currents.

1.2 LCD screen

The LCD screen is a particular output that allows you to write characters on the 16-characters LCD screen, located on the extension board. By examining Appendix B, you will notice that the LCD requires 7 output ports from the microcontroller. On the left side of the LCD screen, there is a potentiometer that allows you to control the contrast of the LCD screen. The 8 first characters of the LCD screen are defined to be row 0, the 8 last characters of the LCD screen are defined to be row 1. The exact mapping of the rows and columns of the LCD screen are shown in Figure 1. The following (partial) list of functions are useful for using the LCD screen:

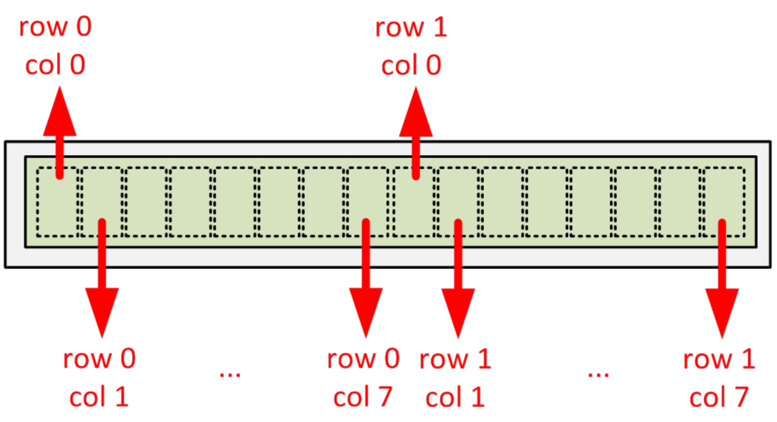


Figure 1: Row and column mapping of the LCD screen.

- `LCD_Init()`: to initialize the LCD;
- `LCD_ClearDisplay()`: to clear the LCD display;
- `LCD_Position()`: to control the cursor position of the LCD display;
- `LCD_PrintString()`: to print a string of characters;
- `LCD_PrintNumber()`: to print the decimal value of a 16-bit value;
- `LCD_PrintInt8()`: to print a two-ASCII-character hex representation of the 8-bit value;
- `LCD_PrintInt16()`: to print a four-ASCII-character hex representation of the 16-bit value;
- `LCD_PutChar()`: to print `char` value;

Exercise 2. Open the `Lab0` project, which has already instantiated the LCD screen. Write a code such that counts the number of times that SW2 is pressed, and prints this value on the LCD screen under the format "SW2 count: 8" (where 8 should be replaced with the number of times SW2 is pressed).

1.3 Keyboard

A small 4 × 3 keyboard is provided, that can be connected directly to the bottom of the extension board[1]. A library has already been created and included in the project, and the keyboard has been

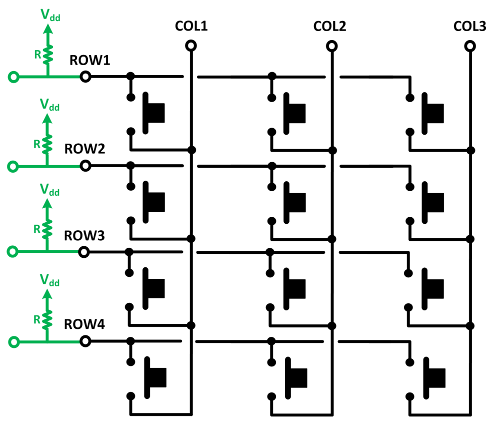


Figure 2: Electronic circuit of a 4 × 3 keyboard. The green part represents the resistive pull-ups inputs of the PSoC board.

instantiated in the `TopDesign.cysch` file. You can use the following function that are included in the library:

- `keyboard_Start()` to initialize the keyboard;

[1]there are 8 pins to the bottom of the extension board to allow for 4 × 4 keyboards. Please use the seven leftmost pins for a 4 × 3 keyboard.

- `keyboard_Scan()` to scan the keyboard. This will return the value that is pressed as a `uint8_t`, and it will return the value `z` if no key is pressed.

Exercise 3. Now write a code that prints successive pressed keys on the LCD screen. Your code should scan the keyboard every 50 ms. Note that if the keyboard is pressed only once, the value should be printed only once on the LCD screen.

2 Introduction to the logic analyzer

The second part of the lab is to help you familiarize yourself with the logic analyzer.

You should start by installing the logic analyzer software, as explained in Appendix C. You can connect the female-to-female cables to the logic analyzer inputs, and male-to-male pin headers at the end of each cable (feel free to detach the cables and to snap the male-to-male pin headers to the amount you prefer).

In the following subsections, you will learn how to operate the logic analyzer:

- running the logic analyzer manually;
- using the logic analyzer trigger.

You can always refer to the logic analyzer tutorial (`https://learn.sparkfun.com/tutorials/using-the-usb-logic-analyzer-with-sigrok-pulseview`).

2.1 Running the logic analyzer manually

We will use the logic analyzer to measure the execution time of parts of our code. The code needs to be instrumented to make internal signals from the microcontroller visible from the outside.

Exercise 4. In this exercise, we will measure the period at which a GPIO is toggled.

- Using the `CyDelay` function, write a code to toggle the first GPIO on jumper J1 of the extension board at a frequency of 20 Hz. There is no need to use a hardware timer for this exercise;
- Connect the analyzer to the extension board as shown in Figure 3.
- Use the `PulseView` software to measure the period with which the GPIO is toggled.

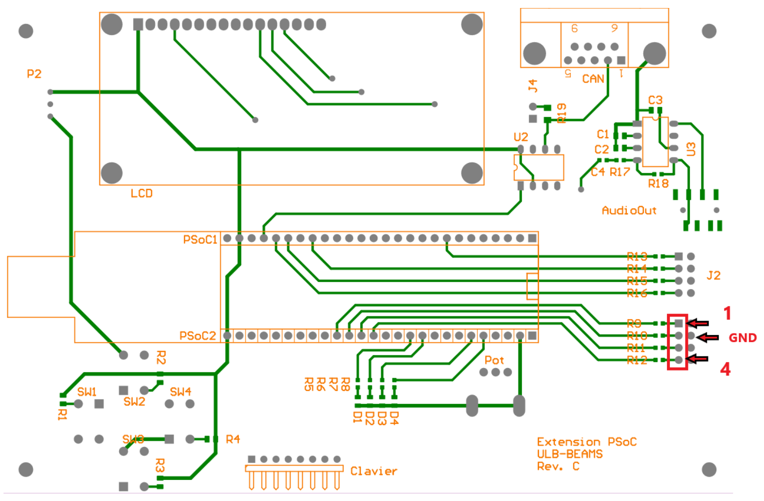


Figure 3: Where to connect the logic analyzer to the extension board.

2.2 Using the logic analyzer trigger

A trigger can be used to automate the acquisition of the logic analyzer. You can select a trigger by selecting the channel to trigger on (e.g. D3) and selecting the required trigger (e.g. rising edge).

It is also possible to trigger the acquisition based on multiple channels, you just need to specify the trigger condition for each channel (e.g. trigger when D1 is high and D2 has a rising edge).

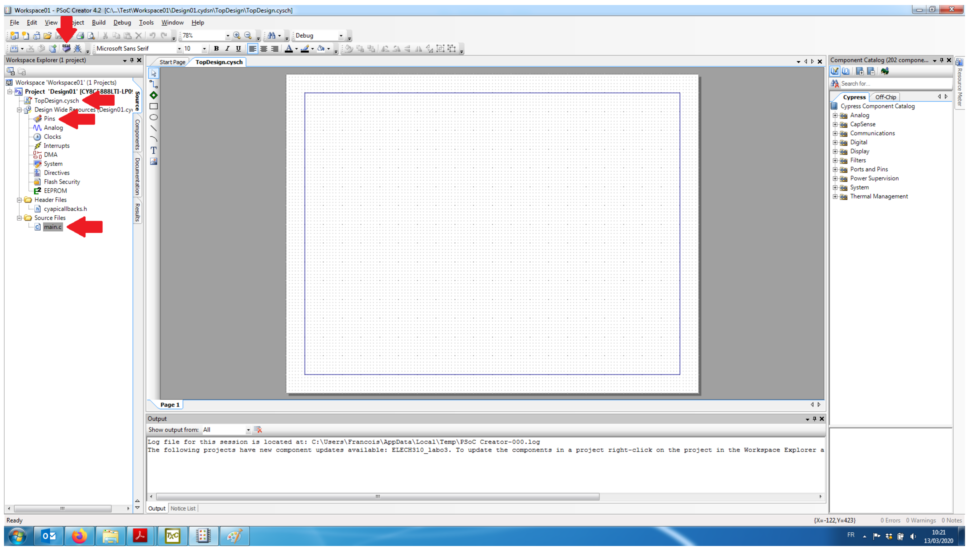
Exercise 5. In this exercise, we will trigger the logic analyzer when SW4 is pressed.

– Extend your previous code such that the second GPIO of jumper J1 becomes high when SW4 is pressed, and becomes low when SW4 is released;
– Specify a trigger condition such that the logic analyzer acquisition is triggered when SW4 is pressed.
– Extend your previous code and trigger conditions such that the acquisition is triggered when SW3 is high and SW4 is pressed.

A PSoC software

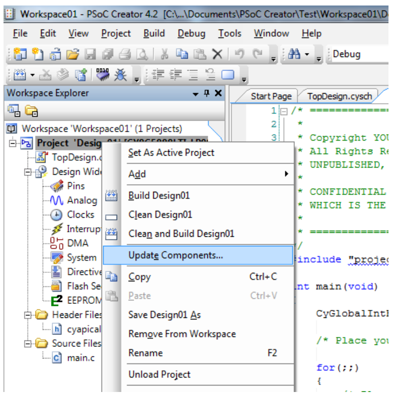
To program the PSoC microcontroller, you need to install the PSoC Creator software suite. You can download this software on the following website:
`https://www.cypress.com/products/psoc-creator-integrated-design-environment-ide`. The software is quite large, so it might take a while to download. You can then install the software suite.
You should also download the CY8CKIT-059 Kit setup on `https://www.cypress.com/file/416376/download` (you might need to create an account to download this file). Once downloaded, you should execute the downloaded file to install the CY8CKIT-059 Kit.

When launching the PSoC Creator IDE, it will usually ask you to register. You can ignore this by selecting "Register later". When creating a new project, you need to specify which microcontroller you are using. In this project, we will use the "CY8CKIT-059 (PSoC 5LP)". Once your project is created, your project will be loaded as shown in Figure 4.



```
Log file for this session is located at: C:\Users\Francois\AppData\Local\Temp\PSoC Creator-000.log
The following projects have new component updates available: ELECH310_labo3. To update the components in a project right-click on the project in the Workspace Explorer a
```

Figure 4: PSoC Creater IDE.



```
/* ============
 *
 * Copyright YOU
 * All Rights R
 * UNPUBLISHED,
 *
 * CONFIDENTIAL
 * WHICH IS THE
 *
 * ============
*/
#include "proje

int main(void)

    CyGlobalIntE

    /* Place you

    for(;;)
    {
```

Figure 5: How to update the PSoC firmware.

In this project, you will mainly be using three tabs of the IDE:

- **TopDesign.cysch:** In this window, you will instantiate all the hardware elements that you will use in the microcontroller, i.e. the GPIOs, the ADC, the DAC, the PWM, the LCD screen, etc.
- **Pins:** This window will help you to associate each hardware element you instantiated to an actual output pin of the PSoC board. The numbers of each pin is written on the PSoC board itself, and you can use the document `Extension_PSoC.pdf` to see which pin of the PSoC board was connected to which input/output of the custom-designed extension board. One of the particularities of the PSoC board is that each hardware element can be connected to any pin.
- **main.c:** In this window you will write the firmware, i.e. the actual program that will be executed on the microcontroller. The code mainly consists of a main function that contains an initialization code and an infinite for-loop. Your microcontroller will continuously execute the program that is written in the infinite for-loop until the microcontroller is turned off.

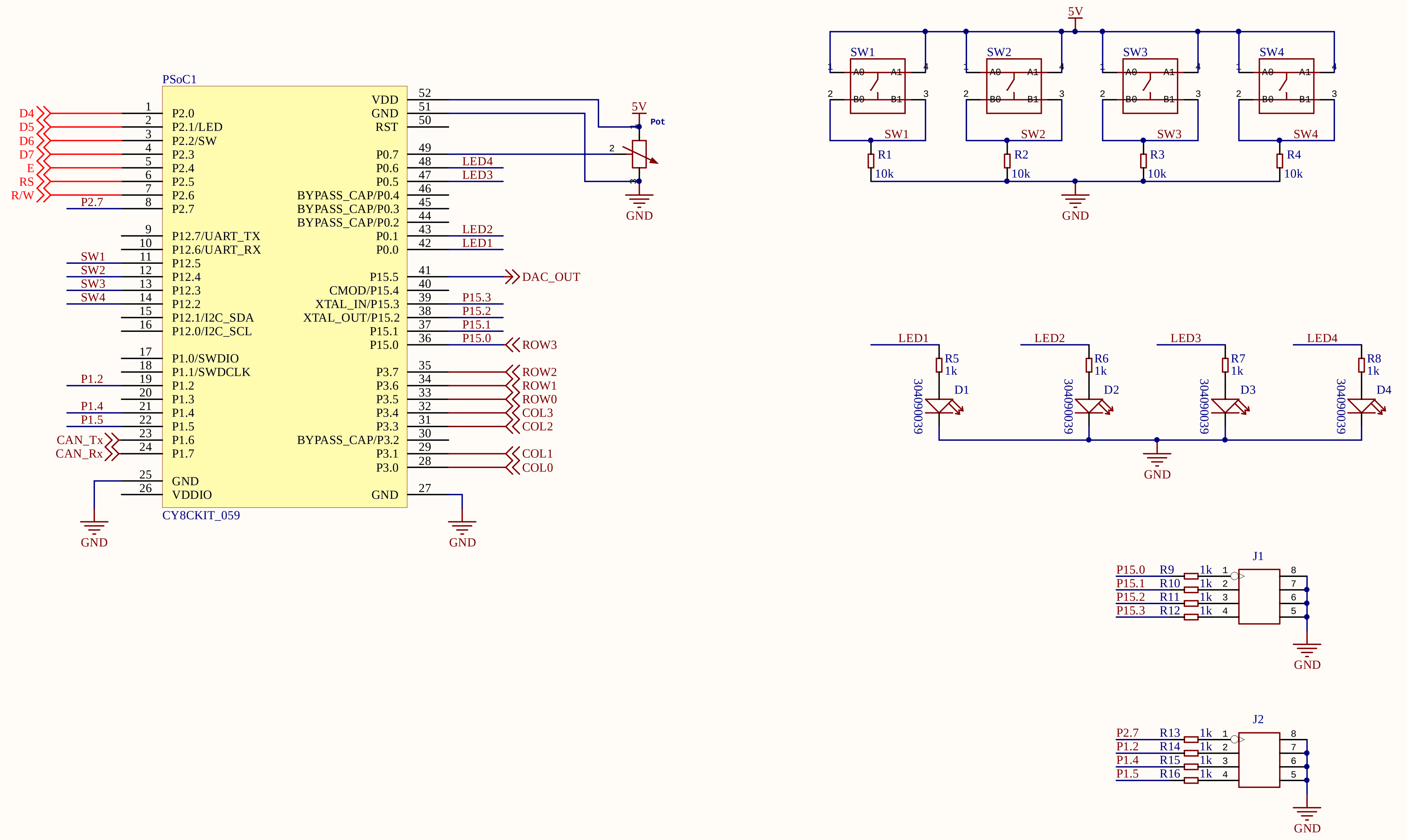
To upload your program to the PSoC board, connect the PSoC programmer to your computer with the USB cable, and click the "Program" button. This will build, compile and upload your project to the microcontroller. Useful documentation can be found on the following websites:

– Getting Started with PSoC 5LP: `https://www.cypress.com/file/41436/download`
– Tutorials for each hardware element of the PSoC: `www.cypress.com/psoc101`
– Video example on how to use the PSoC: `https://www.cypress.com/video-library/PSoC`

It is possible that the first time you try to upload a program to the PSoC board, you get an error message. This is because the firmware on the PSoC board needs to match the software version of PSoC Creator IDE. To solve this problem, right-click on the project and select "Update components" (see Figure 5. Then just follow the instructions with the default settings.

B Extension PSoC

The following pages contain the detailed schematics of the custom-made extension board for the PSoC device.



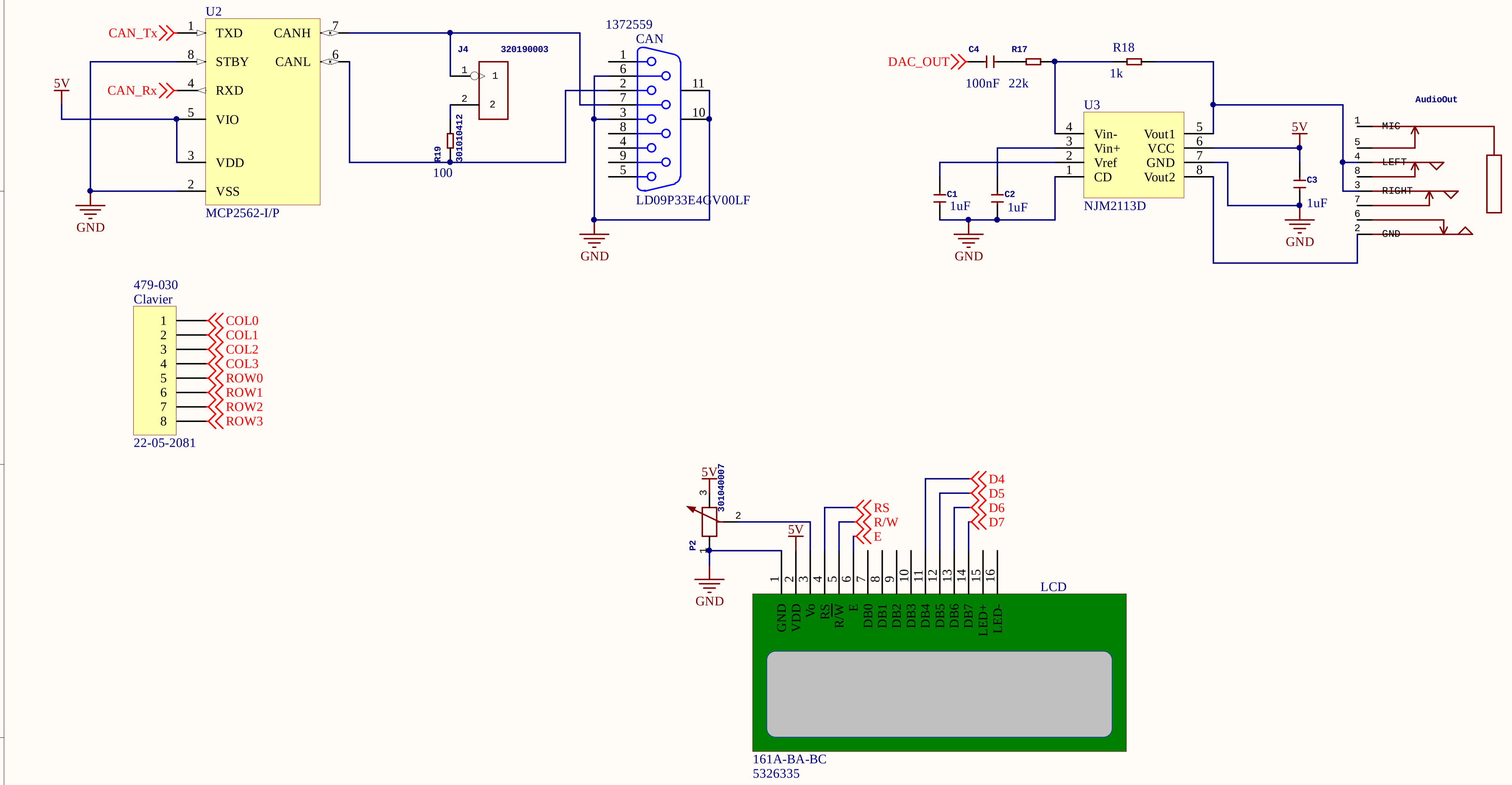
Title *Extension PSoC*

Size: A4 Drawn by: M. Osée Revision: B

Date: 09-01-19 Time: 14:02:08 Sheet1 of 2

File: C:\Users\mosee\Documents\Altium\Projects\Enseignement\extension PSoC\Sheet1.SchDoc

beams

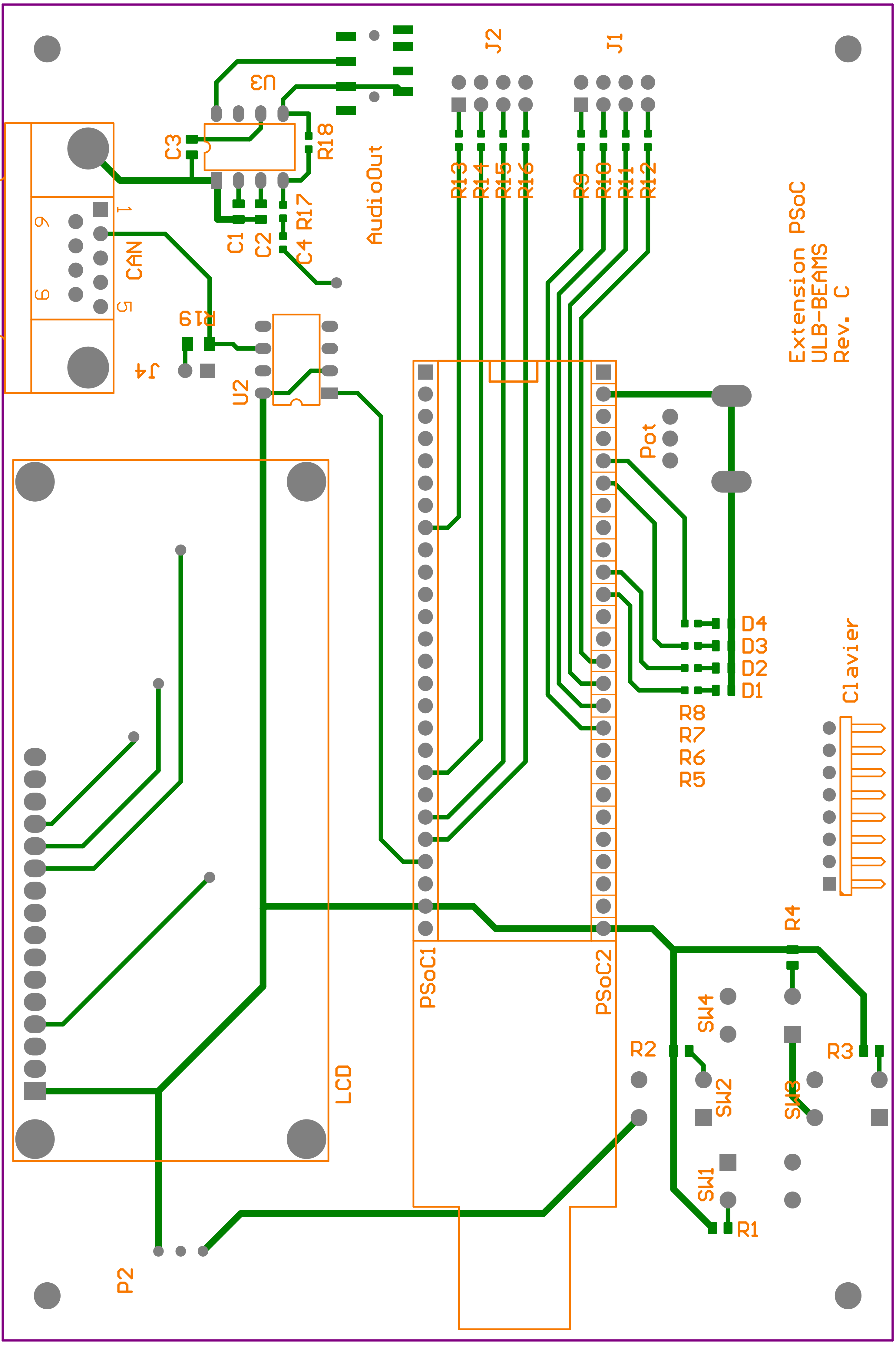


Title *Extension PSoC*

Size: A4 | Drawn by: M. Osée | Revision: B

Date: 09-01-19 | Time: 14:02:08 | Sheet2 of 2

File: C:\Users\mosee\Documents\Altium\Projects\Enseignement\extension PSoC\Sheet2.SchDoc


Extension PSoC
ULB-BEAMS
Rev. C
J1
J2
U3
C3
R18
AudioOut
R13
R14
R15
R16
R9
R10
R11
R12
C1
C2
C4
R17
CAN
1
5
6
9
R19
J4
U2
Pot
D4
D3
D2
D1
R8
R7
R6
R5
Clavier
R4
PSoC1
PSoC2
LCD
SW4
SW3
SW2
SW1
R1
R2
R3
P2

C Logic Analyzer software

To control and operate the logic analyzer, you must install the drivers for the logic analyzer and the `PulseView` software. All of these are available on `https://sigrok.org/wiki/Downloads`, where you need to download:

– `sigrok-cli`, which contains the drivers for the logic analyzers;
– `PulseView`, which allows to operate the logic analyzers and visualize the data.

Download and install both of these programs.

Once both of these programs are installed, connect the logic analyzer to an USB port, and run `zadig.exe (sigrok-cli)`. Click on `Install Driver`. Once the drivers are installed, close the program.

You can now run `PulseView`. If the logic analyzer is connected, it should be recognized by `PulseView` automatically (it should appear as "Saleae Logic" in the connected device). If it's not connected automatically, you should do the following:

– Click on the small arrow next to `Demo device` or `<No Device>`;
– Click on `Connect to Device`;
– In the `Choose the driver` menu, select `fx2lafw (generic driver for FX2 based LAs)`;
– Click `Scan for devices using the driver above`;
– Select `Saleae Logic with 8 channels` and click `OK`.

You should see the 8 channels D0-D7, corresponding to the 8 logic channels of the logic analyzer.