Lab 1: USB to Serial UART Converter Design in KiCAD

COEN-4720 Embedded Systems
Cris Ababei
Dept. of Electrical and Computer Engineering, Marquette University

1. Objective

The objective of this lab is to design the PCB of a USB to Serial UART converter. We will use the popular FT232RL chip as the primary component. While you can buy cheap ones for less than $2 on ebay (with fake chips possible too), here we make our own. In this process, you will:

- Learn to use KiCAD PCB layout tool to make your own PCB (in this lab 1)
- Learn how to solder with a reflow oven (later in lab 10)
- Create something useful that can be used to program for example Arduino mini and many more, beyond the things we do in this course. The converter will serve you for many years to come.

2. Introduction to the FTDI Chip

On UART Serial and RS232 Serial Communication. One of the oldest, but still very popular communication protocol is the RS232 Communication Protocol (or Standard or Specification). RS232 stands for "Recommended Standard 232" and it is a type of serial communication used for transmission of data at medium distances (up to 50 feet). RS232 Serial communication was very popular for transferring data between a computer and peripherals such as printers.

In the RS-232 standard, a logic high ('1') is represented by a negative voltage – anywhere from -3 to -25V – while a logic low ('0') is represented by a positive voltage anywhere from +3 to +25V. On typical PCs that are equipped with RS232 Serial ports, these signals vary between -13 to +13V.

To actually implement the logic/algorithm of transmitting and receiving data according to the RS232 Serial standard a Universal Asynchronous Data Receiver and Transmitter (UART, also called a UART circuit or peripheral or device) is always needed - as shown in Fig.1. UART circuits receive and transmit data serially and this method of serial communication is sometimes referred to as TTL (transistor-transistor logic) Serial because TTL levels between the limits of 0V and VCC (typically 5V or 3.3V).

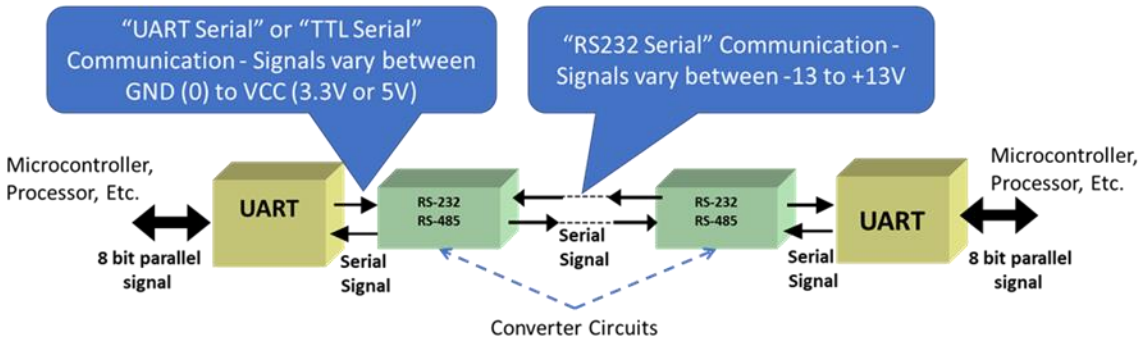

Figure 1: System level diagram of Serial communication.

However, because the UART circuit works with voltage levels GND (for '0' logic) and VCC (for '1' logic), we need an additional component to translate and convert between the two different voltage swings used by the "UART/TTL Serial" and "RS232 Serial" communication from as shown in Fig.1. That component is usually a level converter circuit such as the MAX 3232, as shown in Fig.2.

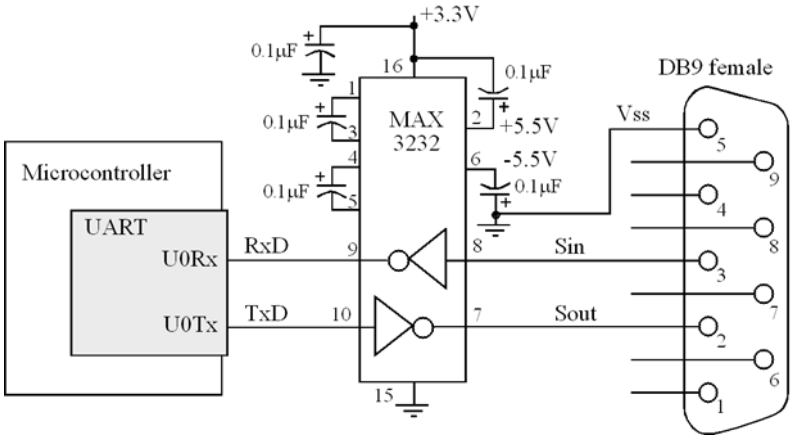

Figure 2: MAX3232 circuit translates and level-shifts voltage swings between UART Serial and RS232 Serial communications.

NOTE: Parts of the discussion in this section were inspired by online articles. For more details please check them out:
https://www.sparkfun.com/tutorials/215
https://circuitdigest.com/article/rs232-serial-communication-protocol-basics-specifications

USB to UART Serial Converter. Now, in many situations we would like to directly connect to the UART device inside say a microcontroller. In this case would connect directly to the RXD and TXD signals of the UART device or peripheral. These signals are connected to GPIOs of the MCU chip and available to access directly on many microcontroller boards.
In these situations, we could connect from our PC, but using a USB port - because the old RS232 Serial ports are not really available on modern computers. In such a scenario, we would use a "USB to UART Serial" adapter (or converter or circuit) as shown in Fig. 3.

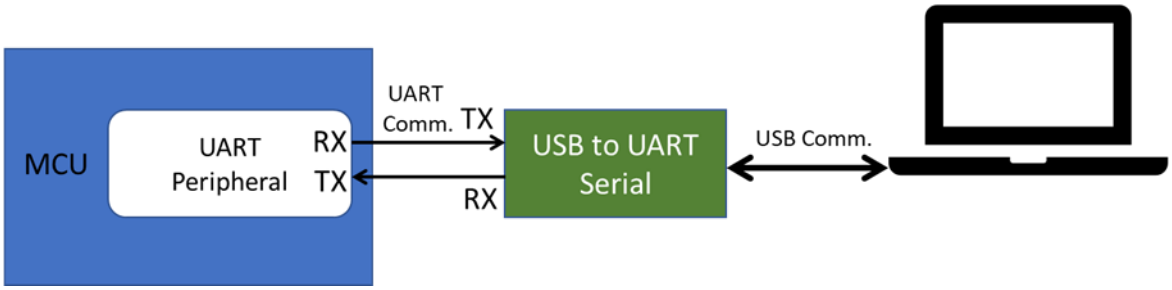

Figure 3: Direct connection to UART device using a USB-to-UART converter.

The FTDI FT232 chip is found in thousands of electronic products, including Arduinos, test equipment, and other electronics. It is a rather simple chip, that converts converting USB to UART Serial port. Because it is so useful, it has been possibly one of the most cloned ICs ever.

We will use one of these chips to create our own converter. Here are a few of its key hardware features:

- Single chip USB to asynchronous serial data transfer interface.
- Entire USB protocol handled on the chip – No USB-specific firmware programming required.
- UART interface support for 7 or 8 data bits, 1 or 2 stop bits and odd / even / mark / space / no parity.
- Data transfer rates from 300 baud to 3 Megabaud (RS422 / RS485 and at TTL levels) and 300 baud to 1 Megabaud (RS232).
- Transmit and receive LED drive signals.
- New 48MHz, 24MHz,12MHz, and 6MHz clock output signal Options for driving external MCU or FPGA.
- Integrated 3.3V level converter for USB I/O .
- Integrated level converter on UART and CBUS for interfacing to 5V – 1.8V Logic.
- True 5V / 3.3V / 2.8V / 1.8V CMOS drive output and TTL input.
- High I/O pin output drive option.
- Integrated USB resistors.
- 3.3V to 5.25V Single Supply Operation.

More details and pointers to sources of information can be found at:
http://www.ftdichip.com/Products/ICs/FT232R.htm

3. KiCAD PCB Layout Tool

To create our USB-to-UART converter, we will use KiCAD PCB layout tool. We will use that to create the schematic diagram of the circuit converter, the layout, and its routing. We'll generate the Gerber files that are needed for PCB manufacturing/fabrication.

Step 1: Download and Install KiCAD

Download and install KiCAD for your specific OS from here:
https://www.kicad.org/download

Please take enough time to read at the minimum the "Introduction" and "Getting Started in KiCAD" tutorials from the learning resources available here:
https://docs.kicad.org/#_getting_started
However, you should read the other tutorials also, especially as you will encounter questions during using the KiCAD tool.
Another great learning resource is "Beginner's Guide to KiCad" from SparkFun:
https://learn.sparkfun.com/tutorials/beginners-guide-to-kicad/all
Yet another good resource is Adafruit's tutorials:
https://blog.adafruit.com/?s=kicad+tutorial
Again, take as long as needed to go through these tutorials and possibly work them as you are reading them. Time put into it now is a great investment for your later skills!

Step 2: Download SparkFun Library

Read about libraries here:
https://www.kicad.org/libraries/download/

Many components available on DigiKey have included links to SnapEDA and UltraLibrarian (Mouser use LibraryLoader, which you need to install on your laptop to then be able to convert the CAD model downloaded from Mouser to say KiCAD format or to download directly by searching for the part in the LibrrayLoader application on your laptop) where we can download EDA/CAD models, i.e., symbols and footprints. For example, take a look at this component:
https://www.digikey.com/product-detail/en/texas-instruments/LDC1101DRCR/296-48297-1-ND/8347716
and search "EDA Models" on the above page to see the links to SnapEDA and UltraLibrarian models available.

However, for our purposes, we only need to download the SparkFun libraries, which we'll use shortly.
---So, download SparkFun Electronics KiCad Libraries from here:
https://github.com/sparkfun/SparkFun-KiCad-Libraries
Download the entire above repository by clicking Code->Download ZIP
Unzip the archive and place it in a new folder called "Libraries_And_Modules_Of_Others" inside KiCAD installation directory, in my case:
M:\KiCad\share\kicad\Libraries_And_Modules_Of_Others\SparkFun-KiCad-Libraries
NOTE:
Search for JLCPCB KiCAD part libraries (https://support.jlcpcb.com/article/84-how-to-generate-the-bom-and-centroid-file-from-kicad)

--Import symbols in KiCAD

1. In KiCad, go to Tools -> Edit Schematic Symbols.
2. Click on Preferences -> Manage Symbol Libraries.
3. In the Global Libraries tab, click on Add existing library to table, the small folder icon and navigate to the downloaded .lib files inside SparkFun-KiCad-Libraries\Libraries. Select all .lib files, then click Open.
4. When you do the above steps for only one component, you can follow up by using the search bar on the left pane of the window to search for the imported symbol and double-click it to open. For example search for "FT232RLSSOP" and then click on it to open it.

--Import Footprints

1. In KiCad, go to Tools -> Edit PCB Footprints.
2. Click on Preferences -> Manage Footprint Libraries.
3. In the Global Libraries tab, click on Add existing library to table, the small folder icon and navigate to the downloaded folder where your .kicad_mod files are located; inside SparkFun-KiCad-Libraries\Footprints. Select those files. Then click OK.
4. Use the search bar on the left pane of the window to search for the imported footprint and double-click it to open.

Step 3: From Circuit Functionality to Schematic Diagram

First, we need to create the schematic diagram of the circuit. This must be done with a perfect understanding of what we want our circuit to do. It's always a good idea to start by looking at the datasheet of the ICs we use. In this case, the datasheet of the FT232R.

We want mainly a couple of things to keep it simple:
1—USB to UART conversion with the ability to output levels 3.3V or 5V. This can be done as shown in *Section "6.4 USB Bus Powered with Selectable External Logic Supply"* of the datasheet:
http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT232R.pdf
2—Let's include also the ability to do to USB to MCU UART interface. For that we need to include a few more outputs to our module board as shown in *Section "7.4 USB to MCU UART Interface"* of the datasheet
3—Definitely, add two LEDs to show RX and TX activity. See *Section "7.5 LED Interface"* in the datasheet.

Step 4: Schematic Diagram Capture in KiCAD

Ok, now we can create the schematic diagram using the schematic capture tool in KiCAD.

Launch KiCAD and create a new project called usb2uart inside some directory. In my case I created it in this directory:
M:\MARQUETTE\COEN4720_EmbeddedSystems_V2\kicad_sandbox
The creation of the project, created the project folder as:
M:\MARQUETTE\COEN4720_EmbeddedSystems_V2\kicad_sandbox\usb2uart
Where all the project files will be located.

Inside the main KiCAD window double click the usb2uart.sch.
This open the schematic diagram sheet, which is empty.
First, let's edit the Sheet Information that is shown in the lower right corner; so, do:
File -> Page Settings
And then, change the Size to US Letter. Enter the Issue Date the date of today. As Title, type USB 2 UART Serial Converter. As company enter Marquette. Click OK.

Ok, time to place symbols on our schematic diagram empty sheet.
Use the place component icon to search for symbols of the following components that we'll use:

--FT232R Chip – Search for: FT232RLSSOP
--USB Type B Connector – Search for: USB_B_PTH (https://www.sparkfun.com/products/139)
--Jumper 3 pins – Search for: CONN_03
--Connector 6 pins – Search for: CONN_06LOCK
--Resistor 10K – Search for: 10KOHM-HORIZ_KIT-1_6W-5%
--Resistor 270K (2x) – Search for: 330OHM-HORIZ-1_10W-5%
--LEDs Red, Green – Search for: LED3MM
--Capacitor 100nF (4x) – Search for: 0.1UF-0402-16V-10%
--Capacitor 10uF – Search for: 10UF-POLAR-EIA3216-16V-10%_TANT_
--Capacitor 10nF – Search for: 10NF-0603-50V-10%
--Ferrite bead – Search for: ferrite

The final schematic should look like the one in Fig.4 below.

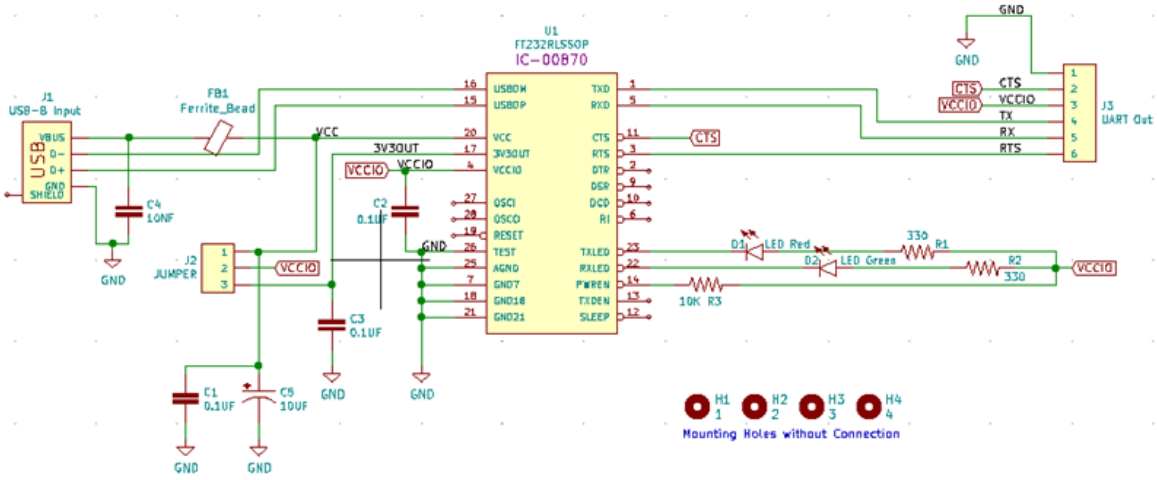

Figure 4: Schematic diagram of the USB to UART converter breakout board.

Step 5: Layout

Once we are done with creating the schematic (double check it for errors! Look at datasheets and application notes), we assign footprints to parts, and design a circuit board!

---Footprint Assignment

Then, assign footprints. Click the Assign PCB Footprints to Schematic Symbols icon. A new window will open up. Assign footprints to all your components.

KiCad has a clear separation between a symbol (the schematic view) and a footprint (the PCB view). To associate symbols with footprints, we need to run **CvPCB**. This sub-program gives us the ability to associate footprints with our schematic symbols. To do that, go down the list in the center of the CvPCB window that contains a list of all the components in our schematic and associate a footprint with each part. Footprints are on the right, libraries (or categories) are on the left. To view the selected footprint in a new window, click the 'view selected footprint' button.

The footprint assignment to symbols is a very tricky and important step because the footprints must be perfect - matching exactly the actual physical component you will buy and solder on the PCB. Some footprints out there are buggy, with errors, and it is really frustrating to manufacture the board with those (that you may find for free online, created by different people) and when it comes to soldering you realize that the footprint does not match the component (which may be too big or too small). Footprints may have errors, so it is a good idea to double check them, create them on your own, or get them from more trusted/verified places like SparkFun or AdaFruit.
This is especially important in the case of components with a lot of different footprints. For example, the capacitor of 100nF (typically used in a pair for power decoupling).

NOTE: Make sure that you add to the footprint library table all footprints you got from different places. In the main KiCAD window to Manage Footprint Library and add the new locations to new footprints!

NOTE: When we run CvPCB it will automatically match footprints that it finds in the libraries. If you know a symbol was downloaded from SnapEDA, check that the footprint downloaded for that symbol is assigned automatically! Though, you can assign any footprint from other library you want! I do that, for example got the symbol from SnapEDA, but its footprint from LibraryLoader because it was cleaner! So, mix and match!

NOTE: It is a good idea to actually check on say DigiKey that the components with the dimensions you use in footprint assignment are available for purchase. If so, then, great your are all set with the footprint assignment for that component. If not, then, search on DigiKey an available component with the same value, which may have different dimensions. In this case, you need to download or search in your footprint library the right sized footprint to assign – in this way we make sure that what we'll buy will fit perfectly the footprints that we use to create the PCB!
By the end of doing the footprint assignment, in this way, we also have checked all the components we need to buy!

At the end of the footprint assignment, you should have assignments as shown in Fig. below.

Symbol : Footprint Assignments

1	C1 -	0.1UF	Capacitors:CAP-PTH-5MM
2	C2 -	0.1UF	Capacitors:CAP-PTH-5MM
3	C3 -	0.1UF	Capacitors:CAP-PTH-5MM
4	C4 -	10NF	Capacitors:CAP-PTH-5MM
5	C5 -	10UF	Capacitors:CPOL-RADIAL-5MM-10MM
6	D1 -	LED Red	LED:LED_3MM
7	D2 -	LED Green	LED:LED_3MM
8	FB1 -	Ferrite_Bead	Ferrite_THT:LairdTech_28C0236-0JW-10
9	H1 -	1	Hardware:STANDOFF_GROUNDING
10	H2 -	2	Hardware:STANDOFF_GROUNDING
11	H3 -	3	Hardware:STANDOFF_GROUNDING
12	H4 -	4	Hardware:STANDOFF_GROUNDING
13	J1 -	USB-B Input	Connectors:USB-B-PTH
14	J2 -	JUMPER	Connectors:1X03
15	J3 -	UART Out	Connectors:1X06
16	R1 -	330	Resistors:AXIAL-0.3
17	R2 -	330	Resistors:AXIAL-0.3
18	R3 -	10K	Resistors:AXIAL-0.3-KIT
19	U1 -	FT232RLSSOP	Silicon-Standard:SSOP28DB

Figure 5: Final footprint assignments.

---Generate Netlist
First, from the schematic, click on the 'Generate netlist' icon.

---From Schematic to PCB

>Click on Pcbnew icon.
>File ->Page Settings; then type in your name and title.

>Then, before anything, define some design rules following OshPark website as explained here:
https://www.youtube.com/watch?v=dM5b_s2ysVk
and manually set the following in your KiCAD tool:
https://docs.oshpark.com/design-tools/kicad/kicad-design-rules/

>Then, read the netlist: click Read Netlist icon.
If all went well, then you should see all you components agglomerated as a cluster as shown in Fig.6 below. Note that footprints are not really placed and connections are not routed either of course. We do manually first the placement and then the routing.

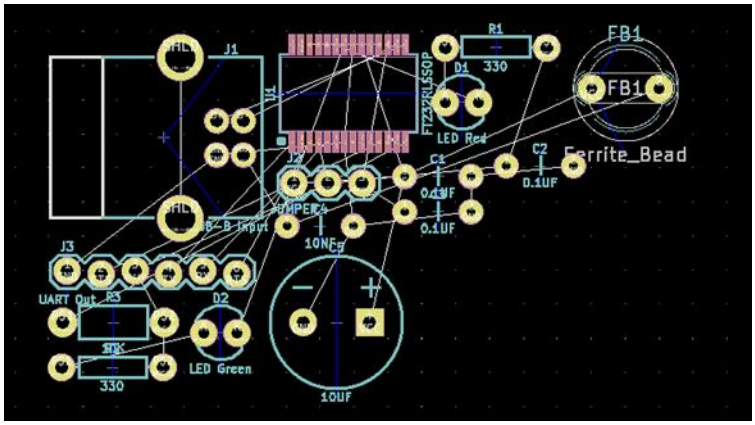

Figure 6: View of all components right after importing the netlist.

KiCAD per-se does not have really any design automation algorithms for placement and routing. The only thing KiCAD does is that it does not overlap all components as it used to be at (0,0); it places them packed but not a real placement. You must do it manually. In time, as you will do more PCB designs, perhaps you will arrive at the same: “PCB design is 90% placement and 10% routing". So, placement is very critical. Do it in a way that minimizes the routing that will be needed later; connections are indicated by the straight lines connecting footprints.

>**Do board outline** (will be actual PCB size during cutting). Just use “Add Graphic Lines” icon in the layout tool. See steps here:
https://www.youtube.com/watch?v=dM5b_s2ysVk (minute 11)
You can edit each of the four lines of the outline with “e”.
Make sure you set line thickness to 0.001 inch and the layer the lines are is Edge.Cuts.

Once done, your placement should look like the one shown in Fig.7.

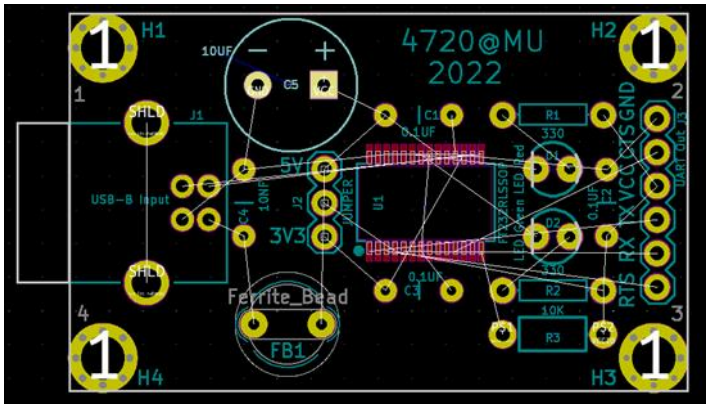

Figure 7: Layout after manual placement.

---Routing

KiCAD does not have its automatic router. While there are ways to use third-party routing tools (requires separate installation of such tools, then, export of layout, routing, export of that routing, and then import back the routing in KiCAD), we'll do manual routing because this is a simple and small PCB.

So, we'll do manual routing. For manual routing, use the following Manually editing a PCB Layout tricks:

Keyboard shortcuts in KiCad's Pcbnew:

+ - Press to switch next layer.
- - Press to switch to previous layer.
m - Move item.
b - Update ground polygon pours.
Delete - Remove a trace or component.
x - Route new track.
v - Add through via.
n - Next grid size. Use with caution. There will be tears if you use a grid outside of 50mils or 25 mils.
Page Up - Return to the top copper layer.
Esc - Escape mode or whatever command in progress and return to normal pointer mode.
ctrl+z - Undo. Use liberally to undo any mistakes.
ctrl+s - To save. Make sure to save often!

NOTE: If you want to use thicker wires, you can set custom Tracks in Setup -> Design Rules -> Global Design Rules
https://techexplorations.com/kicad-4-book/index-p=127.html
I use that for rerouting VCC and GND traces, to make them double thick.

---Add Filled Zones

https://hackaday.com/2016/12/09/creating-a-pcb-in-everything-kicad-part-2/

Click on the 'Add filled zones' button, and a 'Copper Zone Properties' window will show up. Here, you can assign a layer of copper to a specific signal. I do that on the back side and attach it to GND.

---Write a name version and date on the board. Do it on F.Silk layer. To do it click on the "T" icon.

At this time, the two sides of the PCB board should look like in Fig.8.

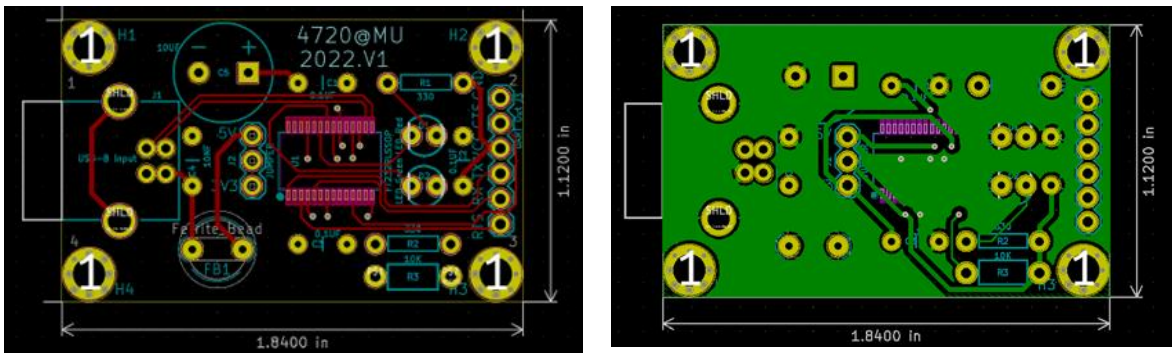

Figure 8: Front and back sides of board.

---Exporting Gerbers
https://learn.sparkfun.com/tutorials/beginners-guide-to-kicad/all
https://support.jlcpcb.com/article/149-how-to-generate-gerber-and-drill-files-in-kicad

Click on the **'Plot'** button next to the printer icon in the top bar to open the *'Plot'* window.

In general, there are 8x layers you need to have a PCB fabricated:

- Top Copper (F.Cu)+ Soldermask (F.Mask) + Silkscreen (F.SilkS)
- Bottom Copper (B.Cu) + Soldermask (B.Mask) + Silkscreen (B.SilkS)
- Board outline (Edge.Cuts)
- Drill file

In the Plot window with the Plot format set for *Gerber*, be sure these *Layers* are checked:

- ☑ F.Cu
- ☑ B.Cu
- ☑ B.SilkS
- ☑ F.SilkS
- ☑ B.Mask
- ☑ F.Mask
- ☑ Edge.Cuts

Additionally, click on **'Generate Drill File'** button. You can use the defaults here as well. Check the "Merge PTH and NPTH holes into one file" box. Click **'Drill File'** or press enter to generate the drill file. Click on **'Close'** in the 'Drill Files Generation' window.
Click **'Plot'** to generate the gerber files for the layers and then click **'Close'**.

---Review Your Gerbers
This is the last chance to catch any errors.
It's always good to review the gerber layers so that you catch any errors before sending to the fab.
Return to the main KiCad project window and open up GerbView by clicking on the button. Once KiCad's GerbView is open, click on **File -> Load Gerber File**. Select all the files shown and click **Open**.

Step 6: Order PCBs

Once we are
Order your PCBs from manufacturers like Oshpark (USA, faster, better quality, more expensive) or JPLPCB (slower, worse quality, cheaper).

---Solder Paste Stencils
Are you doing SMD reflow? Need to order a stencil to apply the solder paste to your board?
Turn on F.Paste in the Plot window to generate the top paste layer.
This ***.gtp** file is sent to a stencil fabricator to create the stainless steel or mylar solder paste stencil.
If you're unfamiliar with stenciling solder paste, SparkFun has a great tutorial:
https://www.sparkfun.com/tutorials/58
For getting stencils created you can use for example:
https://www.oshstencils.com/#%20
We'll not do that in this tutorial.

---Creating Bill-of-Materials (BOM)
https://support.jlcpcb.com/article/84-how-to-generate-the-bom-and-centroid-file-from-kicad
The BOM or Bill of Materials file tells the manufacturer the locations where components are installed. This information will be used during the automated assembly work.
BOM file is a simple text file in comma separated form (CSV).
For now, we can generate the BOM from the Schematic view inside KiCAD; this will give us a list of the components, which I then pair with available components from DigiKey or Mouser.
So, do: Tools → Generate Bill of Materials
Then, use Mouser or DigiKey to find the actual components to buy – make sure you get those with physical dimensions that match your footprints!

Purchase all the components and wait to receive them together with the PCBs! Exciting wait time!

4. Lab Assignment

Reports

For each lab with a "Lab Assignment", you must turn-in an individual typed (not handwritten!) lab report in PDF format (font size 11, single spacing), named **lab#_report_lastname.pdf** (where "#" is the number of the lab, for example in the case of lab1, "#" must be replaced with 1) which should contain the following sections:

- Lab title, Your name
- Introduction section – a very brief description of the problem you solve in this lab assignment, outlining the goal and design requirements.
- Solution section – describe the main technique(s) that you utilized to develop your solution. Include block diagrams, flow graphs, plots, etc. here to aid your explanation.
- Results and Conclusion section – describe your results and any issues you may have faced during the assignment and how you solved them.
- Code listings, C code that you wrote. Appended at the end of your report. Use smaller font (size 9) to save space.

The report should be included into a .zip file that shall include also your project(s) code that you created for the assignment, if any. Name the .zip file as **lab#_lastname.zip**, and upload on D2L in the dropbox folder for the corresponding lab.

For full credit, you must demo the correct operation of your assignment to the TA during the next lab, when that is applicable. You must also demonstrate all provided examples (described in the labs) to the TA for full credit.

Actual Assignment (during the first two weeks of the semester):

Follow through all the steps described in this lab. Create your own PCB layout. In the step described in Figure 8, you must have on your PCB "First-name Last-name 2023" instead of "4720@MU 2022.V1". Send on your own your own PCB files for manufacturing. You can use JLPCB (https://jlcpcb.com/ – in China; may take a few weeks) or Oshpark (https://oshpark.com/ - un US; will take less time to get your PCBs) or any other manufacturer you would like. Purchase on your own all the necessary components too; do it as soon as possible so that you will have time to receive them and possibly correct any mistakes you may make.

In addition to the steps described in this document, please watch the following videos that show how I did it:

1) USB to UART Serial Converter - Part 1: Introduction and Objectives; https://www.youtube.com/watch?v=YGBzahBK7fY
2) USB to UART Serial Converter - Part 2: Install Software and Design Considerations; https://www.youtube.com/watch?v=YuMDFhPcvQs
3) USB to UART Serial Converter - Part 3: Schematic Entry (or Capture); https://www.youtube.com/watch?v=v7kIc4Z1D-g
4) USB to UART Serial Converter - Part 4: Footprint Assignment and Netlist Generation; https://www.youtube.com/watch?v=86rWFbxiwfI
5) USB to UART Serial Converter - Part 5: Placement; https://www.youtube.com/watch?v=sIsJZZ_kQxs

6) USB to UART Serial Converter - Part 6: Routing;
https://www.youtube.com/watch?v=J2sLzAaGzOM

You must write a report to describe what you did, what problems you faced and how you solved them. In your report, you must include your own PCB layout (like the one from Figure 8), which shows your name on it. Upload the report on D2L using the file naming convention: **lab1_report_lastname.pdf**.

5. References and Credits

[1] Textbook:
Carmine Noviello, Mastering STM32 - Second Edition, 2022, Available to purchase online:
https://leanpub.com/mastering-stm32-2nd