Chapter 14 Wk 8 Monday Brd 3: A Golden Arduino PCB

In this week, we start the new board 3 design assignment. This will be a complete, start to finish project to design a “Golden Arduino” board.

We start with the absolute minimum features we need to be able to use this board as an Arduino, to accept uploaded code from a USB port, run the Arduino IDE and be fully compatible with most Arduino Uno R3 shields.

14.1 Purpose of this lab

The purpose of this lab is to gain practice with the entire prototype design flow, including all seven steps. You will have a chance to practice reading datasheets to get useful information, and to realize there is ambiguity in datasheets. Consider adding features in your design to provide options to evaluate design features.

This is a prototype. Be sure to add: test points, indicator LEDs and isolation switches, as needed.

In this week, you will start the POR, identify some of the special, non-commodity parts you will need, and build some prototype solderless breadboard circuits to test out some new circuit elements.

As with all of your boards, use best design practices for routing signals to reduce cross talk (ground bounce) and best design practices to reduce power rail switching noise from I/Os switching.

The commercial Redboard Arduino Uno board in your kit works. It is connected correctly. This means that it meets the ***design for connectivity*** requirements.

But, inspect the board carefully. You will find there are many things you can improve based on what you know about cross talk control and power delivery noise control, bring up and test.

In this board design assignment, you will design your own “Golden Arduino” which meets the same connectivity specs, but has features for better noise control, assembly, test and bring up.

You will walk through each of the seven steps in the board design process.

In wk 8, you will complete your POR and preliminary BOM. This version of the BOM should include just those parts you are concerned about that are not commodity, common parts.

You will not turn this POR in, but you will use it for each step of the design process and include it in the final report.

In the POR, you will sketch out the board design and the risk reduction steps. As a starting place, you should check out some reference designs for Arduino Uno boards, but remember, once you start your schematic, it becomes your design. Do not use a feature from a reference design in your design unless you take ownership of it.

When you get your board back, you will demonstrate the following features:

- *Boot load your Atmega 328 to turn it into an Arduino*
- *Run the Arduino IDE on your board and any standard sketch*

- *You will use your Arduino board to communicate with your brd 4 which will be a multi-sensor shield*
- *Using a special switching noise shield we will give you, you will measure the noise on a commercial board and on your board under identical conditions. Your noise should be 20% to 50% of the noise on the commercial boards.*
- *You will measure the near field emissions from your board, compared to an identical commercial version and find your near field emissions are << 10% that of the commercial Arduino board.*

This is a huge reduction in noise, just from using best design practices!

If you do no other report in your portfolio to show prospective employers, you should post the report on your Arduino board. You will demonstrate that just by using the design principles you have mastered, you have reduced the noise on your board by as much as an order of magnitude over commercial versions of your board. If they hire you, you can do the same for their products. What better way of selling your skills could you come up with?

14.1.1 Schedule

In the ***POR***, you need to be aware of the schedule and the due dates for reaching each specific milestone. Adjust your design and project plan accordingly.

In the POR, be sure to include the items listed in the textbook about the POR. In particular, map the 5-week schedule into your calendar.

Due dates:

Wk 8, POR and preliminary BOM and schematic should be completed

Wk 9, layout is due

Wk 10, CDR of schematic and layout, boards ordered

Wk 13, board 3 comes back, boot load, complete characterization, final due

14.1.2 What you will do in this wk 8 Monday lab

In the POR you should articulate what the purpose of your board is, what it means to "work" and any special features you expect to implement. You will not turn in your POR, but include it in your final report. Your final report for brd 3 is equivalent to your final for this course.

For brd 3, all the parts you will use should already be in the integrated libraries. In board 4, you will learn to create your own symbols and footprints.

Some of the features you will use in your design are:

1. *An Atmega 328 microcontroller*

2. *A CH340g USB to UART interface chip*
3. *A 16 MHz crystal resonator for generating a clock. A 12 MHz resonator for the CH340g.*
4. *Appropriate decoupling capacitors*
5. *A connector for the SPI and boot loading pins*
6. *A TVS chip to protect the data pins from ESD*
7. *Power from the USB plug*
8. *A reset switch with a debounce capacitor*
9. *A 3.3 V LDO*
10. *Header sockets that match the location of the standard Arduino board so that you can plug a shield into your Uno board*
11. *Maximum board size 3.9 inches x 3.9 inches*

Graduate students are also required to add the following features to their Arduino boards:

12. *A ferrite filter on the AVCC pin to the ADC circuit on the 328*
13. *A 0.5 ohm series resistor in the power rail and connectors to it to measure the supply current*
14. *Selectable power from either a 5 V external AC to DC converter or from a USB connector*
15. *Additional rows of ground pins adjacent to and spaced 300 mils center to center from your digital I/O switching pins to reduce cross talk for I/O pins switching.*

Undergraduates can add these features to their board, but they are not required.

14.1.3 Develop and document your POR:

In your POR, you will need to include what it means to "work". These are your functional requirements which will be used to test against your board to give it the acceptance of "yes, it works."

Include in your POR the non-commodity parts you will use. Given the supply chain issues these days, also include price and availability on the JLCpcb LCSC library or on Digikey.

In addition to the schedule, be sure to include in your POR what you will want to implement to reduce the risk and reduce the potential noise, such as:

1. *What special features in the schematic do you want to add to facilitate test and debug?*
2. *What special features do you want to add in the schematic to reduce power rail noise?*
3. *What special features in the parts selection should you consider making your product easier to assemble?*
4. *What special features in the parts selection should you consider to make your product more robust to mechanical stresses?*

5. *What special features in your layout do you want to include to reduce the risk for:*
    a. *Assembly*
    b. *User interface*
    c. *Test and debug*
    d. *Lower cross talk*
    e. *Lower power rail noise*
    f. *Lower cross talk for signals coming off the board through the Arduino pins*
6. *Look at the commercial version of the Arduino Uno board in your lab kit. What three features can you think of doing differently to make your board lower noise?*
7. *What potential risks can you imagine and what can you do to avoid them? Remember, "think like Ralphie's mom."*

Here is one hint: We want to use the same pin out layout for the pin headers so it is compatible with other Arduino boards and shields.

14.1.4 Do not just copy a reference design

You can start your design based on reference designs. Afterall, you are not the first person to design an Arduino board. However, you will see that many reference designs are either wrong or have features which do not apply to your board. In particular, watch out for blindly adding features such as:

- *A MOSFET switch to control the connection between the 7-12 V input and the USB plug*
- *A 5 V on board regulator*
- *A polyswitch resettable fuse in the power line*
- *Multiple value decoupling capacitors*

Here are some examples of reference design to evaluate:

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf

https://www.allaboutcircuits.com/technical-articles/understanding-arduino-uno-hardware-design/

https://learn.circuit.rocks/the-basic-arduino-schematic-diagram

https://www.baldengineer.com/diy-arduino-schematic-checklist.html

http://cdn.sparkfun.com/datasheets/Dev/Arduino/Boards/RedBoard-V22.pdf

https://circuitmaker.com/Projects/Details/Troy-Reynolds/Arduino-Uno-Rev3-Reference-Design

14.2 Step 2: BOM and non-commodity parts

The minimalist board you will design and build will use the Microchip Atmega 328 micro controller. There are multiple ways of implementing the UART to USB interface. We will use the CH340g

You will use the USB mini connector. This is more robust than the micro and a smaller form factor than the large square type A connector.

For powering, you will use the USB connector. If you use an external power jack, implement a feature so that you would NEVER have both connected at the same time.

We will use a 2-layer board for the Arduino Uno. The board will be identical in function as the millions of commercial Arduino Uno boards out there, but you will design and build this one the right way for reduced switching noise.

You will be measuring the noise in the commercial boards and in the one you design and build to see the impact from improving the layout on the noise reduction.

While you can use some of the various reference designs on the internet to start, you will take responsibility for all the design decisions. Just because the design is published does not mean it is a recommended design.

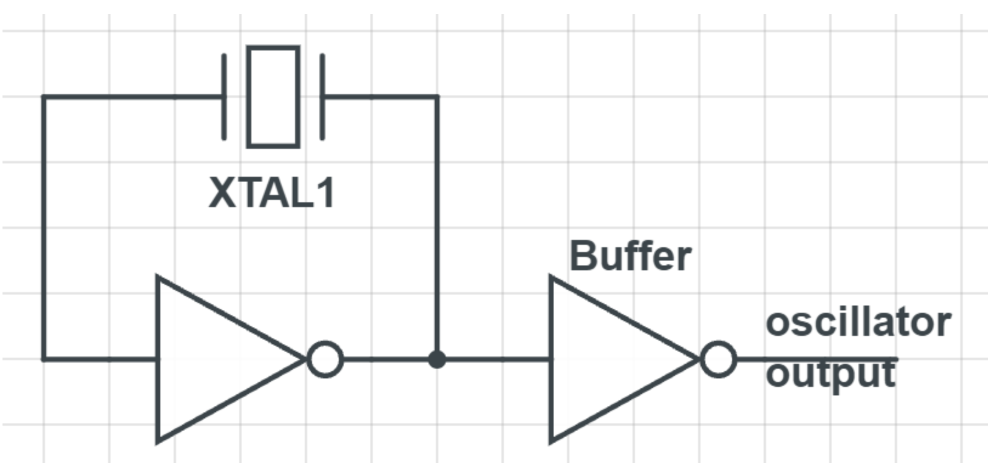
14.2.1 Crystal circuits

There are multiple ways of generating a clock signal on a micro controller. They all follow the common practice of placing a resonating structure across the input and output of an inverter circuit. The inverter will oscillate at the resonant frequency of the resonator.

There are two common types of resonators, quartz crystals and ceramic resonators. The quartz crystal is more stable but has a larger form factor. A ceramic resonator is smaller, and is a little less stable, but still can have a stability within 10 ppm.

For our boards, we will use ceramic resonators. They are smaller size, lower cost and readily available.

In principle, a simple inverter circuit with feedback between its output and input will self-oscillate. The frequency depends on the propagation delay between a change in state at the input causing a change in state at the output which changes the input, etc. Here is the basic circuit:

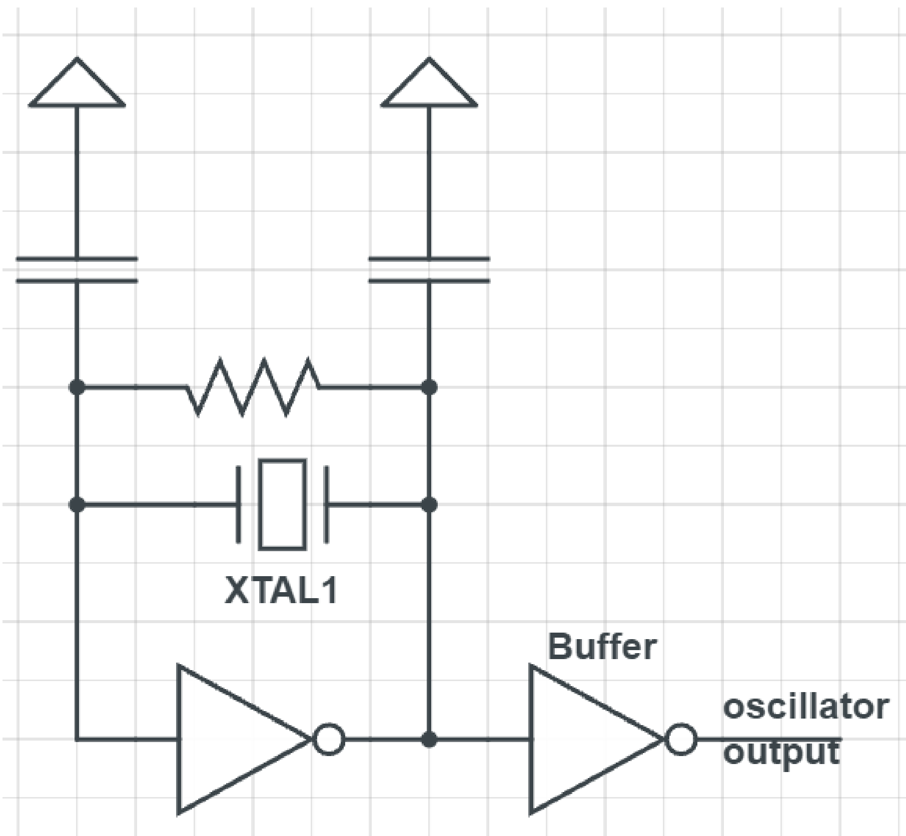


Only one inverter is needed to generate the oscillations. However, it is good practice to add another inverter at the output of the oscillator to buffer the oscillator. This way if the output is loaded down with a low impedance, like 1 k ohms, the buffer will drive the load without impacting the inverter on the oscillator.

In principle, this circuit will drive either a crystal or a resonator. In practice, we need to add two features to make the oscillations more stable and robust.

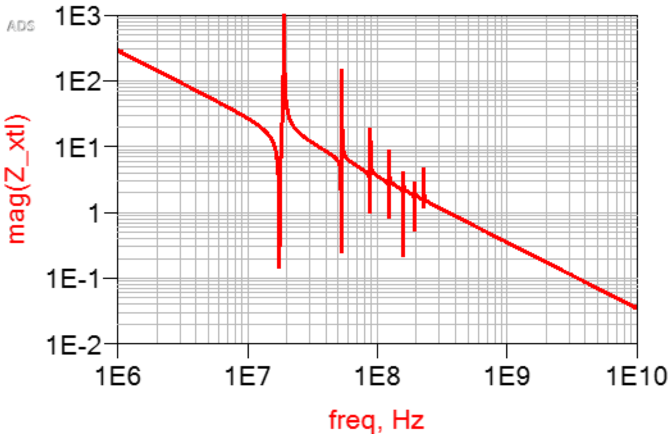
First is a feedback resistor to start the circuit into oscillation. This can be in the range of 1 M to 10 k. With the resonators selected for your board, we have found a value of 10 k works best. For many crystals, 1 M works best.

Secondly, we need to suppress some of the higher mode frequencies that can resonate, so that only the lowest frequency oscillates. This means we need to add two filter capacitors on either side. Their value should be in the 7 pF to 30 pF range, depending on the resonator. Values of 22 pF work well for our applications. This practical circuit is shown below:



14.2.2 How they work

A crystal is a slice of quartz between two electrodes. It is piezo electric. A voltage across the end faces causes a mechanical compression and a mechanical compression of the crystal causes a voltage across its ends. Due to this coupling of mechanical motion and electrical signal, it has an impedance which dips to a low value at the resonant frequencies for which it will vibrate. The figure below is an example of the impedance profile of a typical crystal.



Note that this crystal, as is typical, has multiple resonances above the first harmonic.

While we are only showing the impedance profile, a crystal is not a passive component. It generates a voltage when it is mechanically stressed. At the resonant frequency, the combination of the impedance and voltage response generates a low impedance. The frequency of the low impedance drives the oscillation of the inverter.

When the crystal is in the feedback loop of a high gain inverting amplifier, the frequency where the impedance is lowest, ie, where there is more feedback, and at which there is a high voltage generated by the compression of the crystal, will be the frequency of oscillation.

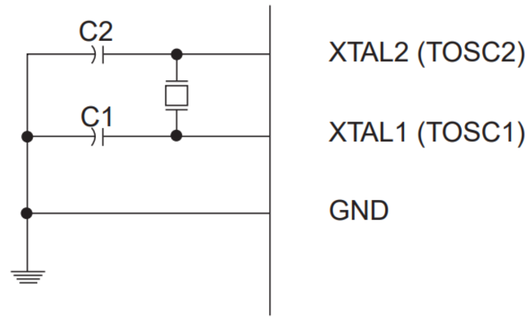
But, the crystal can oscillate at multiple harmonics. In order to use a quartz crystal in a clock generation circuit, we need to implement two important features:

1. *The inverter sometimes needs to be kick started by adding some feedback between the input and the output. This way, the initial transitions of the inverter will drive high frequency components, some of which will overlap the resonant frequency of the crystal. The frequency component with the largest voltage component and the strongest feedback, i.e., the lowest impedance, will be the oscillation frequency. Using a high resistance resistor across the input to output of the inverter will "**kick start**" the inverter into oscillation.*

2. *We need to suppress the higher frequency components of the crystal so that only the first harmonic, the lowest frequency, will drive the resonance. We do this by adding small filter capacitors on either side of the inverter to filter out the higher frequency components at which the crystal might oscillate.*

In the 328 schematic, (Figure 9-2), shown below, there is no feedback resistor specified. There are just the 22 pF capacitors to suppress higher order modes.

Figure 9-2. Crystal Oscillator Connections



The 10k resistor to jump start the oscillations is optional. In our lab experiments, sometimes the resonator self oscillates with no problem when connected to the 328. Sometimes it does not. This is one of those cases where you will want the option of adding the 10k resistor. You can decide if you want to take the low-risk path and add it to your board, or just place the pads for it so you can add it to your board if you do not get oscillation without it.

If you plan to add it, you may want to use a 1206 part so you can manual assemble it to the board.

The traces from the crystal pins of the 328 to the crystal and the capacitors are sensitive to noise and trace capacitance. **Try to route the traces as short as practical**. Try to avoid passing the signal traces from the IC pins to the crystal pads and 22 pF filter capacitors through vias, or routing over gaps in the return path.

In wk 8 Wed, you will be building a solderless breadboard version of an oscillator circuit.

14.2.3 Debounce circuits and switches

There is a reset pin on the 328, PC6. The reset pin is normally held high. This means the reset is off. When the reset pin is momentarily pulled low, the 328 resets the sketch it is running and starts from the beginning.

In a typical application, we will pull the reset pin low in two ways: with a manual switch so we can reset the 328 any time we want by pushing a button, and automatically from the CH340g device. The DTR pin on the CH340g chip will pull the reset pin low when it receives an incoming message from the USB port and has code to transmit to the 328.

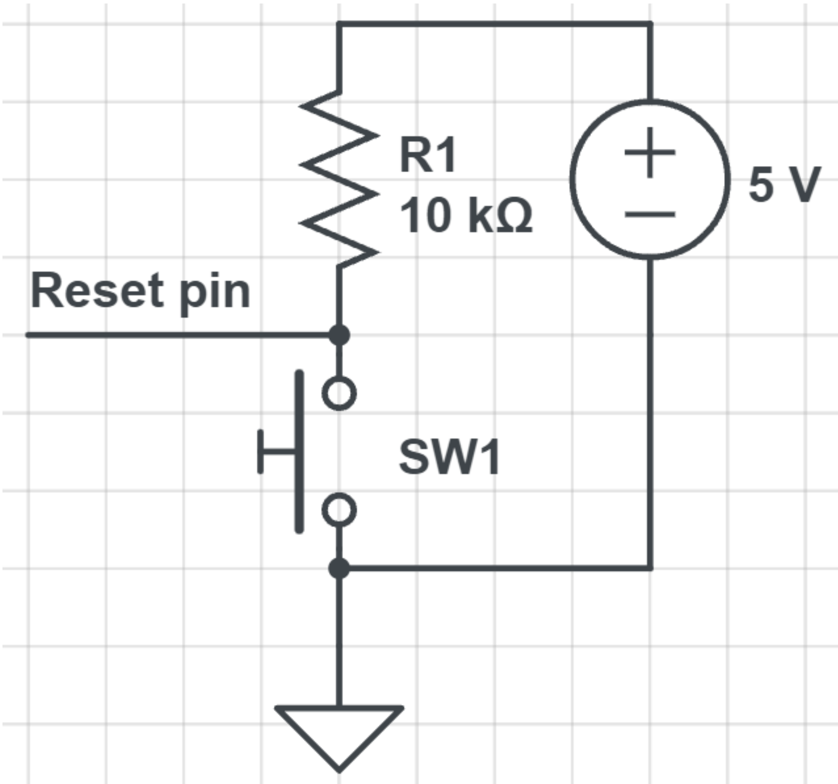
We need to set up the reset pin on the 328 to accept a pull down signal either from the manual switch or the CH340g DTR pin pulling it low. Here is how we will do it.

A mechanical switch we commonly use as a reset is this one:

https://jlcpcb.com/parts/componentSearch?isSearch=true&searchTxt=C174049

When it is pressed, a pair of contacts short to each other.

As an example, the figure below shows a simple pull down switch circuit. The reset pin is normally pulled high, until the switch is closed. Then the rest pin is pulled low while the switch is depressed.



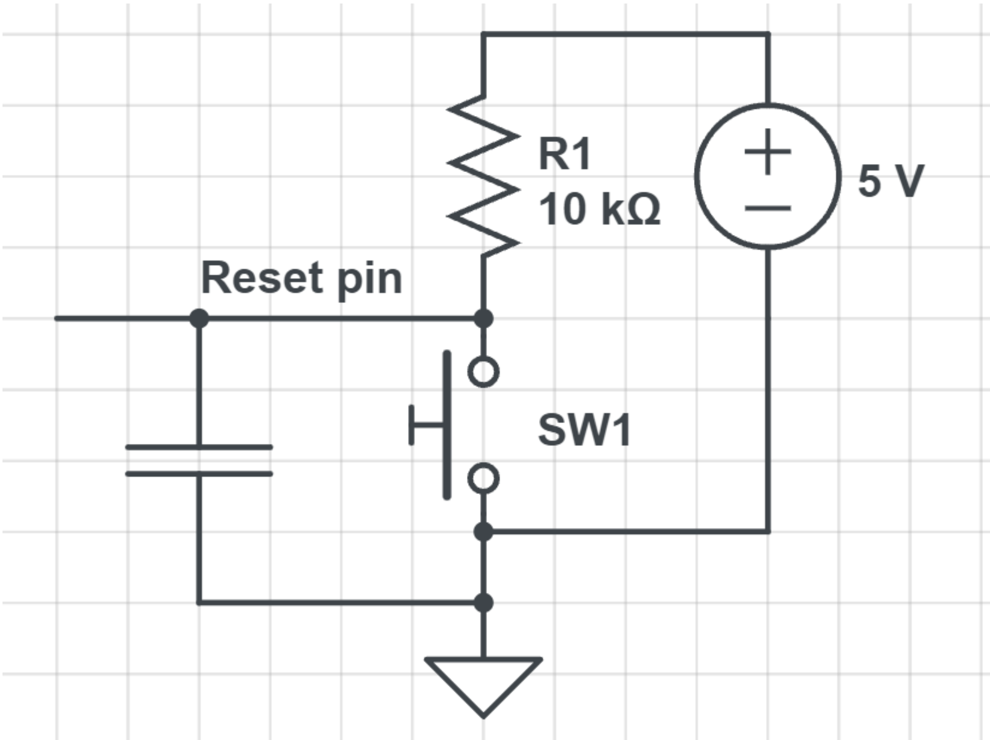
There is one problem with this circuit. Many mechanical switches, after they make contact, “bounce” up and down a few times before they finally stay closed. This is called bouncing. As a little background, check out this article I wrote about bouncing circuits.

If the bouncing happens on a reset pin, it is possible the multiple contacts will cause multiple resets and potentially a fault condition. The typical spec for a resent pin is that a pin has to be brought low for > 1 usec for the reset pin to read the reset pin going low.

A reset pin would have seen this voltage fluctuate many times and go through multiple reset attempts in the 1 msec time before the switch voltage stabilized.

While this can be fixed in software by adding a delay after a reset is detected before another reset is detectable, it can also be fixed in hardware by adding a debouncing circuit.

The purpose of a debouncing circuit is to hold the rest pin low, after the switch is pressed, and until all the bouncing has stopped. This is as simple as adding a capacitor across the switch, as shown in the figure below.



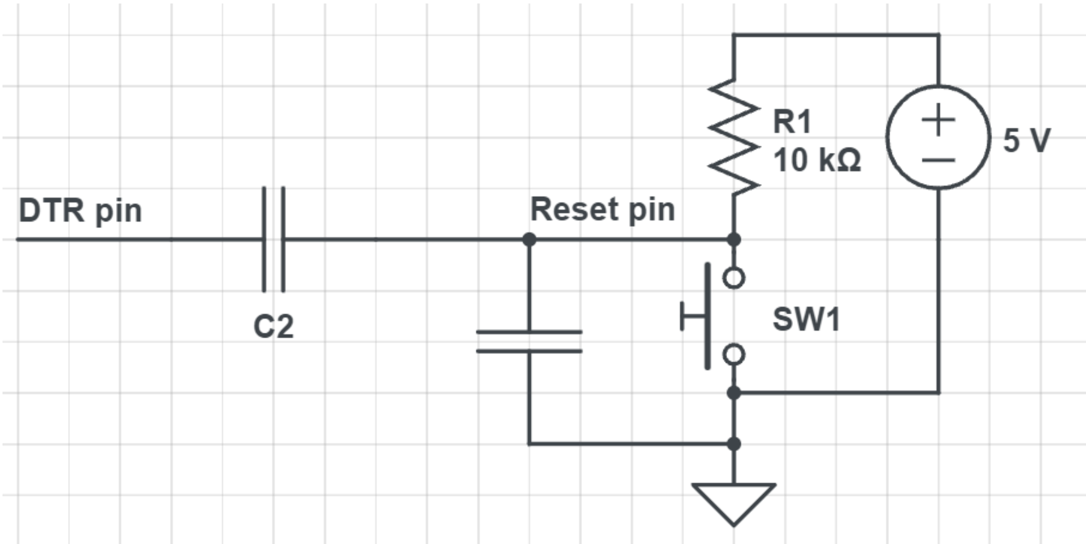
In this circuit, the resistor pulls the reset line high normally. The capacitor is charged to 5 V as well. When the switch is closed, the reset pin is pulled low and the capacitor is discharged to 0 V as well. When the switch bounces up the first time, the capacitor holds the reset pin low. It has an RC charging time, back up to 5 V, of R x C. If the bouncing time for the switch to settle is short compared to the RC charging time, the reset pin will be kept low.

For example, if C = 1 uF and R = 10 k, the RC time constant is 10 msec. As long as all the bouncing time is finished in a time short compared to 10 msec, the reset pin will be kept low. It will not see the bouncing.

Any capacitor > 0.1 uF would be a suitable debounce capacitor. Since the exact values are not critical, use component values you are already using on your board. For example, a 1 k ohm resistor and a 1 uF capacitor.

This is for the debounce circuit.

For the DTR pin, we need another circuit. The DTR pin may be pulled down for a long time. We want to use the initial falling edge to pull the reset pin down. We connect the DTR pin to the reset pin with a high pass filter- a series capacitor. This will only let through the high frequency, negative, falling edge. This circuit is shown below:



The one problem is that we have also built a capacitor voltage divider circuit, The voltage on the reset pin is the voltage divider of the C2 coupling capacitor and the debounce capacitor. If C2 = Cdebounce, the voltage on the reset pin will only be ½ the voltage swing of the DTR pin. It may not be enough to pull the reset pin low.

In order to pull the reset pin low enough when the DTR pin pulls down, C2 should be much larger than the debounce capcaitor. It this is 1 uF, then the C2 should be at least 22 uF.

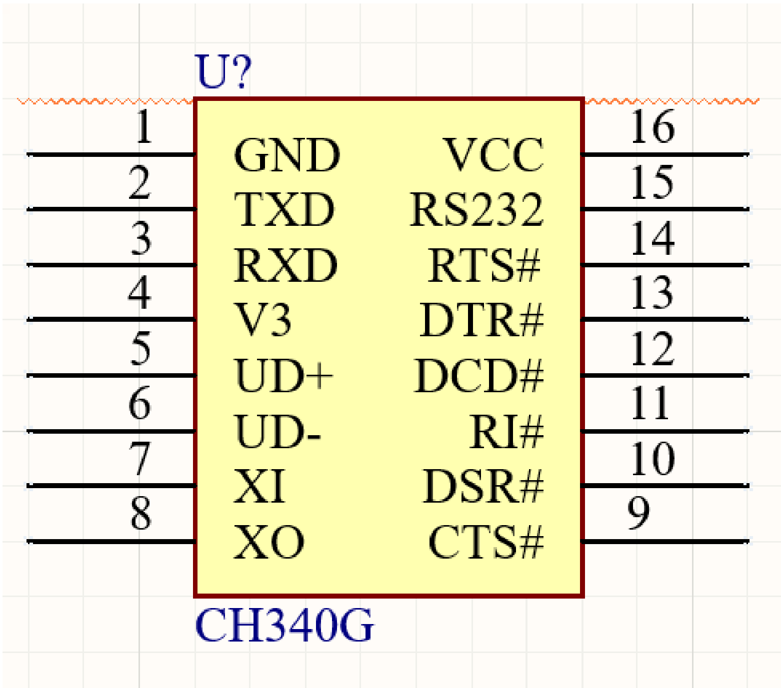
If the DTR pin has a 3.3 V signal range, then when it goes low, the reset pin will only go as low as 5 V – 3.3 V = 1.7 v. This is not low enough to pull the reset pin low. We need to pull the reset pin below 0.8 V.

Make sure the voltage suppling the CH340g chip, that powers the DTR pin is 5 V and not 3.3 V.

There are lots of important design details to pay attention to.

14.2.4 The USB to UART chip is the CH340g

The CH340g is the interface between the USB port and the UART to the 328. The symbol is here:



It connects via the TXD and RXD pins. Be sure to note that the TXD on the CH340 connects to the RXD on the 328 and the RXD on the CH340 connects to the TXD of the 328.

The datasheet for this component, translated poorly from the Chinese, is here:
https://cdn.sparkfun.com/datasheets/Dev/Arduino/Other/CH340DS1.PDF

You will want to power the Vcc at 5 V. However, there is an internal 3.3 V pin, which connects to an internal 3.3 V regulator. This pin is pulled out so you can connect a decoupling capacitor to it. Remember, regardless of what the datasheet says about the values of the decoupling capacitors to use, you know the right way of decoupling.

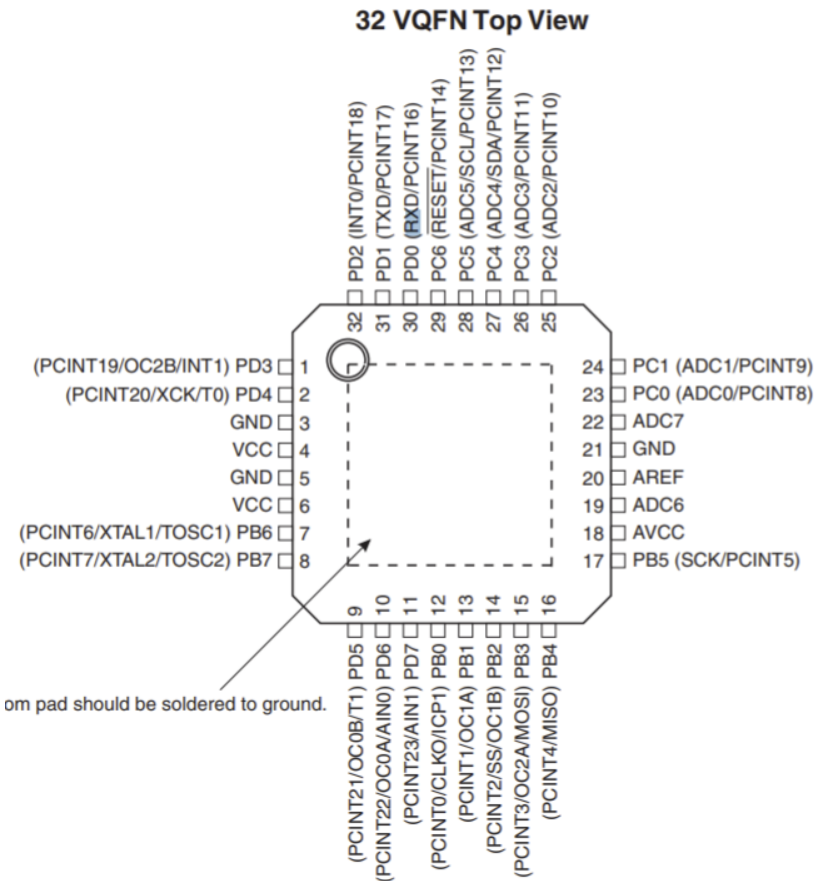
14.2.5 The ATmega 328 microprocessor

A commonly used 328 uC is this one:

https://www.digikey.com/product-detail/en/microchip-technology/ATMEGA328P-ANR/ATMEGA328P-ANRCT-ND/2774230

Take a look at its specs. Check out some reference designs that use this uC as well.

The device we will use in this board is a 32 pin package. The pin out is in the datasheet and shown here:



Note, the RXD pin of the 328 is PD0 and the TXD of the 328 is the PD1 pin.

You can put indicator LEDs on the TX and RX pins using a 1k series resistor and LED. What will this circuit look like?

Use a separate connection to the AVCC and VCC pins. They each get their own decoupling capacitor. Ideally, you should add an LC filter between the power rail and the AVCC pin to keep noise on the Vcc rail from getting onto the AVCC rail. This is with a ferrite inductor.

14.2.6 Using a ferrite bead to filter noise from the PDN to the AVCC pin of the 328

This topic was covered in the textbook in chapter 14.9.

What ferrite is available in the library? What pole frequency will you be able to achieve with this ferrite and a large decoupling capacitor?

What other questions do you have about the components and the circuits you will build?

Chapter 15 Wk 8 Wed: complete the schematic for the Golden Arduino board

In this week, you will complete the schematic design of the Golden Arduino Board.

Note, you will be using the 2021-08_ integrated libraries for your schematic.

The ***POR*** has been completed. It is a living document. As you learn more about the project, you should periodically update the POR. Most important, you should have identified the important risk sites, and mitigation strategies. Part of the POR is the schedule, budget and power requirements.

The initial ***BOM*** was completed, identifying the most important, non-commodity parts. At the same time, you should have reviewed their datasheets.

This week, you complete the schematic. This should include:

- *All non-commodity parts*
- *All other parts*
- *Their connectivity*
- *Including, the power distribution path, test points, LED indicators, isolation jumpers*

In the next week, you will turn the schematic into the layout.

There is value looking at reference designs to get you started, but always take responsibility for your own design. You can review the reference designs mentioned in the last section.

Be aware that there are features in other designs you WILL NOT use and layout principles you SHOULD NOT follow. Take responsibility for your own design.

Normally, in your designs, you would

1. *Find the parts you want to use on digikey*
2. *Download the CAD models- schematic symbol and layout footprint, into a library package project in Altium*
3. *Compile the library package as required, into an integrated library, which will be automatically installed into your board project.*

However, in this project, you will only use parts in your schematic that the assembler we will use, JLCpcb, has in their library. To start your design, use the latest integrate library package. (Note, this is a good starting place for other future designs that might use these parts.)

Some parts in this library are identified as having an LCSC part number. This means they are in the JLCpcb library.

All the parts in the JLC integrated library are available from JLC and can be assembled by them. Choose these parts if at all possible. You should only need to grab the 10x probe test pads from the manual library.

When you select a ***surface mount*** part to be added to your schematic, wherever possible, be sure to select a part that has an LCSC part number and in particular, a Basic type. We are only allowed 10 extended parts in any board, but we can have many more basic parts.

Since JLCPCB will be doing the assembly, it is really ok to use an 0402 part, especially if it is available as a basic part.

If you need a 2 uF part, but we only have a 1 uF part and it is a basic type, figure out a way to use the 1 uF part if at all possible.

Select parts as needed from your integrated library and place them in your schematic.

Place parts on your schematic page with some flow: power on top, signal flow from inputs to outputs, and components used together in specific functions, in close proximity.

Connect up the terminals in the schematic as required. Use net names when appropriate. Think about which ones you want to connect by direct connection vs by common net names.

Try to complete the schematic as well as you can on your own, without consultation except about using the tool.

Create a pdf copy of your schematic and post it on the canvas drop box. This is due on the Monday at the end of the week.

By the end of wk 8, your schematic should be complete

In week 9, you will experiment with a solderless breadboard circuit to test out:

- *A crystal oscillator*
- *A ferrite filter*
- *A reset circuit*

Then you will complete the layout.