COSC 4P78 – Lab 1

Introduction:
Your lab demonstrator should start off with an introduction to the lab environment and rules.
For your convenience, a quick summary of the important parts:

- Don't screw around
- Don't leave the door wedged open
- Don't screw around
- Be extra-careful when the soldering irons are on, even if you're not the one soldering
- Don't screw around
- Most components don't have spares, so feel free to ask for help (or just a second look) before soldering or applying electricity
- Don't screw around

Being a danger to yourself or others will lead to you being promptly ejected from the lab.

Besides that, remember that these labs are for you to try something new. The final requirements will always be given, but you have a reasonable degree of flexibility in terms of how you achieve that. If you want to do something differently, it probably won't be a problem. Just clear anything particularly unusual with your demonstrator, to verify that the basic requirements are being met (and that it's safe for the hardware).

Your Task:
Your only task for this week is to create a 'hackduino', capable of running off of a 9V battery, and of communicating over a TTL serial cable. Everything else that follows will just be advice on one (but not the only) way to achieve that.

Soldering Tips:
Your demonstrator should first go over basic soldering, so this will be a bit redundant.
When you solder two metals together, you're simply using heat to melt a soft metal (the solder) so that the liquid metal can flow into the gap between the two harder metals (e.g. copper) and then resolidify, creating a semi-strong connection (both physical and electrical).

Ideally, capillary action will help a lot with the solder flow. Here's the part that matters:

- You're heating the metal, which heats the solder

If you try to just heat the solder itself, then congratulations: you can easily melt it. Except it isn't likely to go where you want it to, and even if it does it won't form a very good connection. Instead, you want to heat the metal from one or both of the components being joined.

- Make sure you don't heat the components too much. If it isn't doing what you want it to, it's always an option to lay off the heat for a while, and try again
- Once some of the solder gets into place, if you need to add a bit more, you can heat *the solder that's already there*, to use it to melt the solder wire
- Large globs of solder don't really add much benefit, and increase the chances of accidentally shorting two connections together

Definitely feel free to experiment on some scrap wire before even touching the components.

Voltages:
Depending on your background, this might be irritating to read, but your circuit designs will rely on positive voltages (usually +5V). So, if you wish to hook up, say, a light, then you'll need to decide which lead gets the +5 (or Vcc), and which gets the 0V (or Gnd).

Polarized Components:
Some components are polarized. That means they have a defined +'ve and -'ve side. In this case, -'ve goes to ground. For electrolytic capacitors (the little metal cylinders with two wires coming out the same end), the -'ve has a stripe, and is labelled with -. For LEDs, the light has a flat side, marking the -'ve. For many polarized components (including all those used in this lab), the +'ve lead (the *anode*) is longer than the -'ve (the *cathode*).

Breadboards and Protoboard:
Each lab station has a solderless breadboard. It's a small plate with many holes in the top. When a wire (or component lead) is pressed into a hole, it makes contact with the metal inside. The breadboard has the contacts arranged in an interesting fashion:
The left and right sides each have a +'ve and -'ve rail. For each rail, all holes along the vertical rail are electrically connected to each other. Note that the two +'ve rails (and the two -'ve rails) are not connected to each other unless you connect them yourself.
Horizontally, each row has two bands of 5 holes. Each of those 5 holes are also connected, though the two bands are not connected to each other.
There are only 5 breadboards for the lab (one per station), so don't leave components in the breadboard when you leave the lab.

Perfboard is a prototyping board that has a grid of equidistantly-spaced holes arranged across them, often with copper pads on at least one side to allow for easy soldering. Your lab station should have a single piece of 'permaprotoboard': a perfboard with holes, silkscreening, and connections identical to a solderless breadboard, making it easy to transfer a layout from a breadboard into a perfboard, and solder it to create a permanent module.
Note that the pads are on both sides of the board *and* that both faces are connected. This means you can place components on both sides and have more options for connections.

Arduino Background:
A microcontroller is a microprocessor that includes additional components (e.g. RAM). They often store and run just a single program (as opposed to the multitasking dynamos that are traditional computers). The Arduino is just one platform (actually, branded set of platforms) with particular electrical specifications.

We won't actually be building Arduinos. Instead, we'll be adding enough components that we can use an ATmega328p with the Arduino IDE, and program it as though it were a 'real' Arduino.

Test Setup:
The ATmega328s have been preloaded with a simple program that just blinks an LED connected to General Purpose Input/Output (GPIO) Pin 13, by the Arduino labelling, which means Pin 19 on the chip itself. Make sure to put a resistor between the LED and ground, or you'll burn out (at least) the LED.

Serial Cable:
You'll be learning more about TTL Serial communication later, but here's what you need to know for now:
Each lab station has an FTDI cable with a connection capable of providing a transmit (TX), receive (RX), Vcc, and Gnd connection (as well as a couple others we don't really need). When using the cable for communication, the TX for the cable needs to connect to the RX on the ATmega, and the RX for the cable needs to connect to the TX on the ATmega. Additionally, the GND should connect to the ATmega's GND. If Vcc is connected between the two, it can power the board. However, **when doing so, the 9V battery must not be plugged in!**

Power Supply:
Actually, now's a good time to bring up the primary power supply: the 9V battery. However, you can't power the ATmega directly with 9V, so you'll need to bring that down to 5V. You've been provided with a 7805 *voltage regulator* for just that purpose. You should also use an electrolytic capacitor (22uF) and possibly also a ceramic capacitor (22pF) to smooth the output.

Stripping Wires:
Make sure your demonstrator remembered to show you how to strip wires. You'll need to make several patch cables to connect components on the board.

Guides:
There are two guides that are pretty good. It is highly advisable to follow one of them.
This one is particularly good, and there should already be a printed copy at your lab station:
https://www.arduino.cc/en/Main/Standalone

It also includes a pinout diagram, since pin #s on the actual ATmega chip will not match the pin designations in the Arduino IDE (once we get to it). They also include links to the ATmega328 datasheets, but I think that's likely overkill.

This one is also pretty good:
http://www.instructables.com/id/Perfboard-Hackduino-Arduino-compatible-circuit/?ALLSTEPS

Finally, I'm fond of this little cheat sheet, which should also be at your lab station:
http://tinkerlog.com/2009/06/18/microcontroller-cheat-sheet/

Feel free to follow either guide, or a different one, but consider the following tips:

- Don't bend the pins to the ATmega! They're immensely fragile, and you could ruin your only processor for all of the labs! If you're at all uncomfortable or uncertain, ask for help!
- You'll probably want to at least arrange all of the parts onto the breadboard, just to plan out where everything will connect. You may or may not wish to power it while on the breadboard (sometimes it can be a bit hard to keep everything plugged in)
- You've been provided with a socket for the ATmega, but you're absolutely not obligated to use it. Even if you do, I wouldn't bother plugging it into the breadboard
- I'd suggest soldering a 6-pin male header to the board, and connecting the relevant pins to the ATmega, so you can easily plug the FTDI cable in and out
- There should be a little space on the protoboard below (or above) the chip, where you can make additional connections. If you find yourself running out of space, just run a lead to one of those sections
- I've already assembled one and left it in the lab. You're more than welcome to take the layout as one possible suggested configuration

Beyond that, some pin-specific suggestions:

- Pin 1 is the *reset* pin. When that pin is *brought low* (forced to 0V by connecting to ground), the ATmega resets. Obviously, this means we want it to normally *not* be low. For safety, we force it *high*. We can do this with a *pullup resistor*
    - A pullup is a (relatively high resistance) resistor that connects a pin to Vcc, so the voltage is kept high (instead of 'drifting' to unpredictable values). Use a 10K ohm resistor to connect Pin 1 to Vcc
    - You'll also want a *reset* button. Just use a button (momentary switch) to connect Pin 1 to Gnd. When the button is pressed, that'll connect the voltage source to the sink, and force the pin low
        - Make sure to leave yourself enough room for the button, or you can clip off two of the pins from the momentary switch if you prefer. Personally, I'm partial to really secure components when they get pushed around a lot

- Pin 2 is RX, while Pin 3 is TX. Keep this in mind if you want to set up an FTDI header elsewhere on the board
- Pins 2–6 are GPIO pins 0–4
  - Note that this means some pins can be retasked to other purposes
  - You may want to consider adding a female receptacle to make these pins easier to connect to later on
- Pin 7 is Vcc, and pins 8 and 22 are Gnd
  - You should probably just connect them to the +'ve and -'ve rails, respectively, and then connect the 7805 (and FTDI header, if you're using one) to the rails.
- Connect a 10uF electrolytic capacitor between the Gnd and Input lines of the 7805 (do you know which lines are which?).
  - As mentioned above, a 22uF electrolytic and a 22pF ceramic capacitor are good for connecting the output line with the ground, but to make this far easier: just use them to connect the +'ve and -'ve rails directly
    - Make sure the two capacitors are connected in *parallel* to each other
- It's also highly advisable to add a power light
  - You've been given a green LED. Put a resistor (e.g. 1K ohm) between its anode (long leg) and Vcc, and connect its cathode (short leg) to Gnd
- Pins 9 and 10 are for the oscillator
  - The crystal isn't polarized, so don't worry about which way is which. Just use the crystal to connect the two pins
  - You also need two 22pF ceramic capacitors to connect each pin to ground
    - This can be a tricky thing to do. Feel free to use some longer leads, to move some of it to another place on the board, or to use both sides of the board
    - If you decide to do a silly setup like mine, make sure to avoid contact between other components and the oscillator's metal casing
- There are some other groupings that might warrant female receptacles as well (e.g. pins 11–14, 15–19, and 23–28)
- For the AVcc pin, you could just connect it to Vcc, or you might want to try using a couple 180 Ohm resistors (in parallel) to connect it to Vcc
- If you haven't already done so, I'd connect the left +'ve rail to the right +'ve rail, and the left -'ve rail to the right -'ve rail

As mentioned in the test setup above, if you get everything wired up correctly, then connecting an LED to GPIO pin 13 (package pin 19) with a resistor and applying power (whether from the battery or the serial cable) will let it run the sample program of blinking the light.

If you don't get it all finished today, that isn't a huge problem, as next week's lab should be relatively simple. Just make sure you don't leave any components in the breadboard (as the other lab will need it).