Lab 2: Inputs, Outputs, and Time

Overview:
For this lab the bulk of your time will likely be taken up coaxing basic functionality from the LCD display, which has a specification document which may be a bit less than forthcoming when it comes to precise details of what you should do to the GPIOs to get it running. Don't worry too much about that, that's by design.

ALSO: Inside your system there is a scheduler. It is slicing time into fragments and allotting it to programs. Using a system with a scheduler in it requires that you be mindful of times when your program is not running.

In this lab, in addition to expanding the capabilities of your system, you will perform a test of scheduler jitter. To do this, you'll need to set aside one GPIO pin to act as your system monitor. Once you do that, you'll need a task that makes a voltage go low and high. It should use the system sleep command to wait some small amount of time (1us perhaps) and toggle the pin up and down. You will need to show the result of this on the oscilloscope and characterize in some way (peak? RMS? Histogram?) the “scheduler jitter” created by the imperfections in the scheduling process.

You will be required to show this waveform to your TA at demo time and explain it.

Talking to an LCD Display:
The LCD display requires an initialization routine to be followed. This takes the form of flipping some pins up and down in an order as specified by the timing diagrams in the spec sheet. Your LCD will have a blinking cursor on it to prove that your initialization code is working. This is a good point to stop and make sure it is correct before you continue. You won't be able to test anything until the initialization is correct. Debugging this may require a lot of navigating of segmentation faults, as file pointers make it possible/easy to fprintf to a null pointer.

Once you have initialization working, you will need to send text to your display. You should set this up as a pipe that accepts text and pushes it to the display. You may want to research the mkfifo command and how named pipes work. This program will need to be running independently of the program which interfaces it.

It might be a good idea to also have other pipes that can be used to perform special tasks. Another good idea might be to interpret newline, clear, and return characters correctly. Maybe use it as a text mode display for some games?
Talking to Oscilloscopes:
The Oscilloscope is one of the most fundamental tools of EE. It is also pretty handy for low-level embedded. Getting a signal on one of these machines is not difficult, but interpreting it correctly can require some work! If you haven't seen one in a while, talk to someone who has! The good news is that reading a steady digital signal is fairly easy once you actually have that signal. It might be a good idea to verify you have a “pulse” by plugging in an LED. Note that above, oh, 20Hz it will just look like a solid color, so you should slow down the signal while checking. Then it’s just a matter of getting the scope to display a waveform correctly. The lab has some spare oscilloscope probes but it’s a good idea to have your own.

Deliverables:
At least two programs, one that talks to the LCD only and one that does anything else. Full source code for each, including make files, README, etc. NOTE: learn to use one of [make clean] or [.gitignore] before doing this! A full report detailing work done:

No specific format is required, however, it IS required that your lab work be NEAT, LEGIBLE, and communicate what you have done clearly and effectively. You should note that the lab assignments won’t always follow this that well, so don’t take them as a role model. Think of them more like "a client we should probably stop working with after this". There should be clear diagrams which effectively communicate all aspects of the design. There is a standard for diagrams (UML) which might be worth looking at, but if it comes down to "communicate clearly" vs "follow UML", pick "communicate clearly" every time. Note that nowhere in this document are the words "pad your intro with meaningless words". While there is no minimum length, keep in mind that it is very unlikely that you will have enough space to communicate a full accounting of your design and its functions in fewer than five pages. The TAs have been asked to grade you based on whether you do a good job of showing what you have done to them in a way they can read and understand, and whether the presentation is clear and professional.

Demo:
With every lab you will need to demo your working code. Pull the code from your repository and walk through the code with your TA. You can expect your TA to ask you difficult questions about design choices or even attempt to break your project by changing the test code. Make good arguments for your implementation of your design. In this lab you will also need to show that you are able to monitor a userland-code generated pulse with the oscilloscope.