Lab

Part 0: Setting up your Launchpad

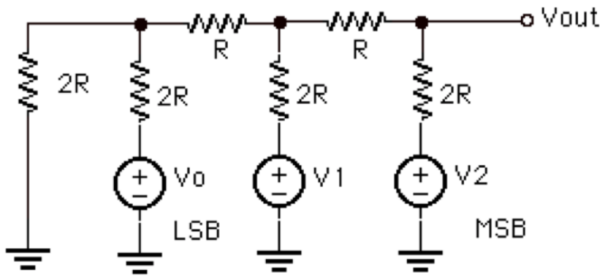
If you do not recall how to set up your Launchpad or you'd just like a refresher, consult Appendix A of this note.

Connect your Launchpad to your computer, open Energia, and upload the `Blink` example file to your Launchpad. Then, test your Launchpad by pressing the reset button (`RST`) on your Launchpad.

> In general, ALWAYS press the reset button after you upload any code to your Launchpad!

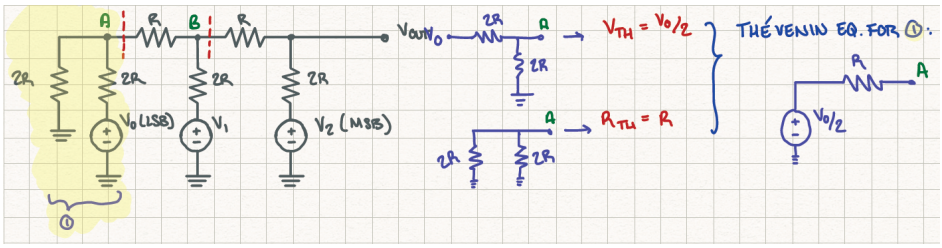
Part 1: Digital-to-Analog Converters (DACs)

We will first build a 3-bit DAC to convert a **binary** input into an analog voltage, and then we will extend it to 4 bits using the knowledge you gleaned from building the 3-bit DAC. The binary input will come from the Launchpad's digital I/O pins while the analog voltage will be probed using an oscilloscope.



We can build a DAC using only resistors in a structure called the **R-2R ladder** (shown above). This structure takes an n-bit binary input and converts it to an output voltage. The bits here are represented by the voltage sources V_0 through V_2. You might have seen this structure in EE16A, but we will analyze it again today. Here is a quick review of superposition and Thévenin equivalent circuits to help you get started:

- **Thévenin's Theorem:** *A linear, active, resistive network containing one or more voltage or current sources can be replaced with a single voltage source and a series resistance.*
- To find R_{TH}, short out the voltage sources and find the equivalent resistance between output and ground.
- We suggest breaking the circuit up along the red dotted lines and working your way from each bit to the output to determine the contribution of each bit to V_{OUT}. *Hint: this may seem laborious at first, but you may notice a pattern that saves you a lot of work.* We have also done the first equivalent circuit for the LSB for you to help you get started.



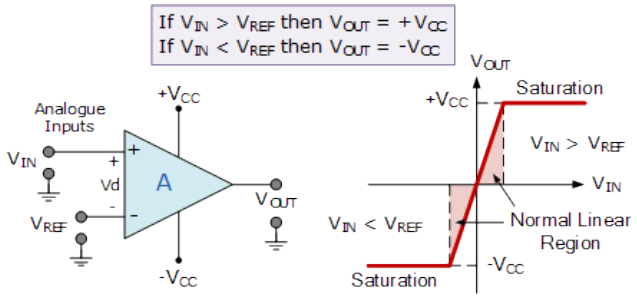
> **Now, complete Part 1 in the Jupyter notebook, starting with the questions.**

Part 2: 4-Bit Analog-to-Digital Converter (ADC)

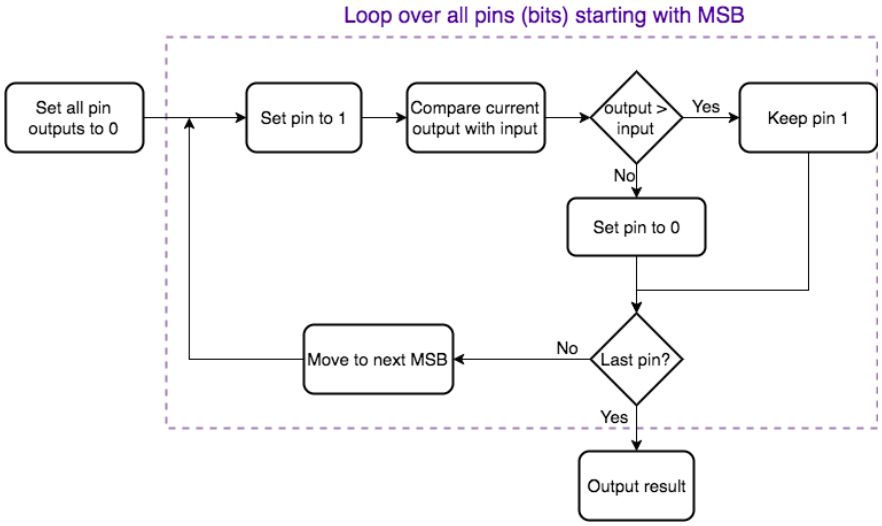
In this part of the lab, we will build a **Successive Approximation Register (SAR) ADC** using the 4-bit DAC you built in the previous part.

Given an analog voltage, an ADC measures it in the digital domain. Digitization of a signal involves both **discretization**, i.e., we restrict its timesteps to being finitely small (infinitely small timesteps $\Rightarrow$ continuous time), and **quantization**, i.e., we fix a finite "set of states" that the signal can have at any given moment, which in this case is the set of integer values between 1 and $2^4 - 1$, inclusive. One commonly used circuit architecture for analog-to-digital converters is the Successive Approximation Register ADC (SAR ADC), which is what we'll be using today.

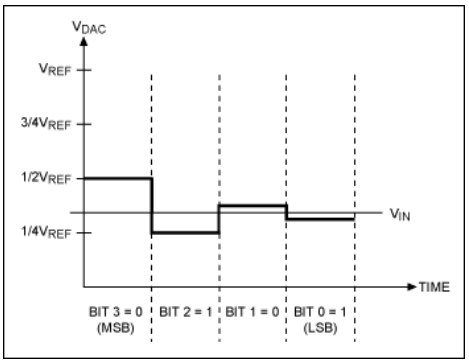
Before we get into the operation of the SAR ADC, we will briefly review the operation of a comparator. A comparator is simply an op-amp configured in open-loop, so that the output saturates at either the positive or the negative rail depending on whether the voltage at the noninverting terminal of the op-amp is greater or less than the voltage at the inverting terminal.



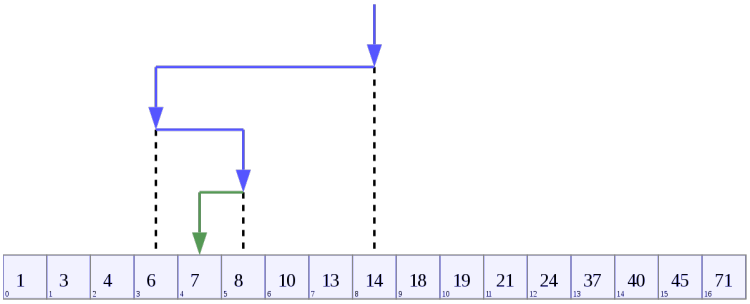
The SAR ADC algorithm tries various binary "trial codes" by feeding them into a DAC to generate voltages and comparing the result with the analog input voltage using a comparator. It then uses feedback to adjust the DAC voltage to get as close as possible to the analog voltage. Here is an illustration of the algorithm. Note that "pin outputs" refers to the MSP outputs connected to the DAC from Part 3, and "current output" means the output voltage of the DAC.



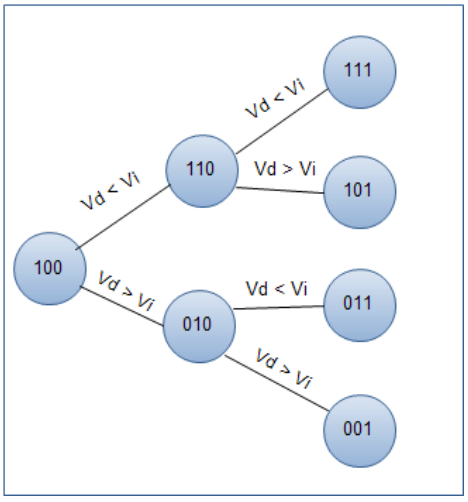
The images below demonstrate how the SAR ADC algorithm is analogous to the binary search algorithm you may have seen in CS 61B.



(a) Output of SAR ADC



(b) Representation of binary search



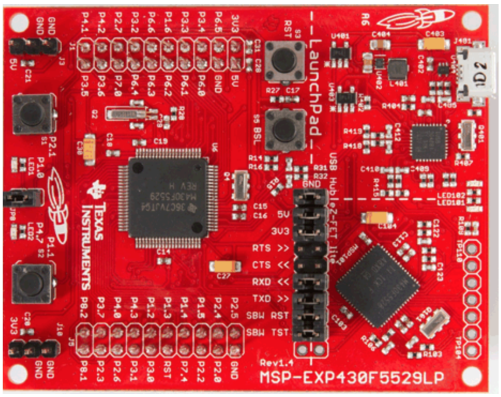
(c) Representation of 3-bit SAR ADC algorithm with binary "trial codes"

Note the similarities between the SAR ADC output in figure (a) and the demonstration of binary search in figure (b). You should be able to see something that looks like a variation of figure (a) when you probe the output of your ADC circuit with the oscilloscope, though the exact waveform will depend on which analog voltage V_{IN} you send into the ADC.

Now, proceed to Part 2 of the Jupyter notebook and follow the directions there.

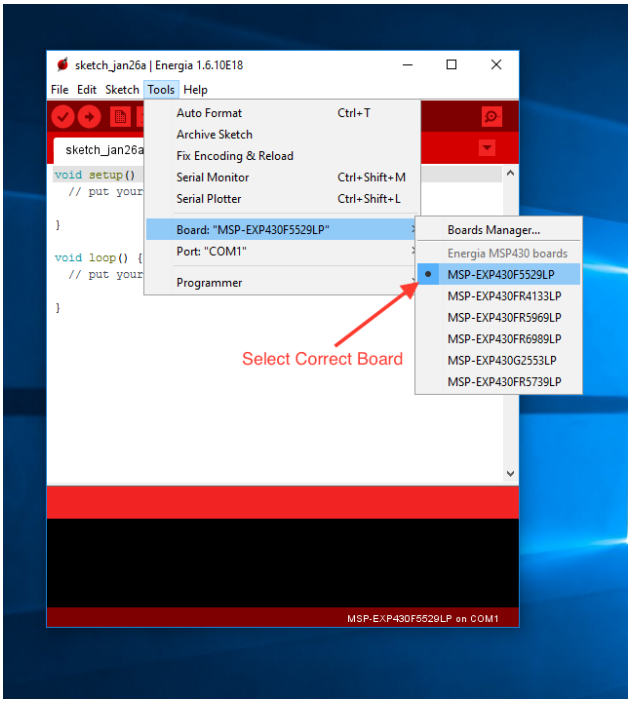
Appendix A: Getting Started with your Launchpad

This semester we will be using the TI MSP430F5529 Launchpad for our labs. The development board includes the MSP430F5529 microcontroller. This part might be a review, but it's good if you need a refresher on working with your Launchpad.

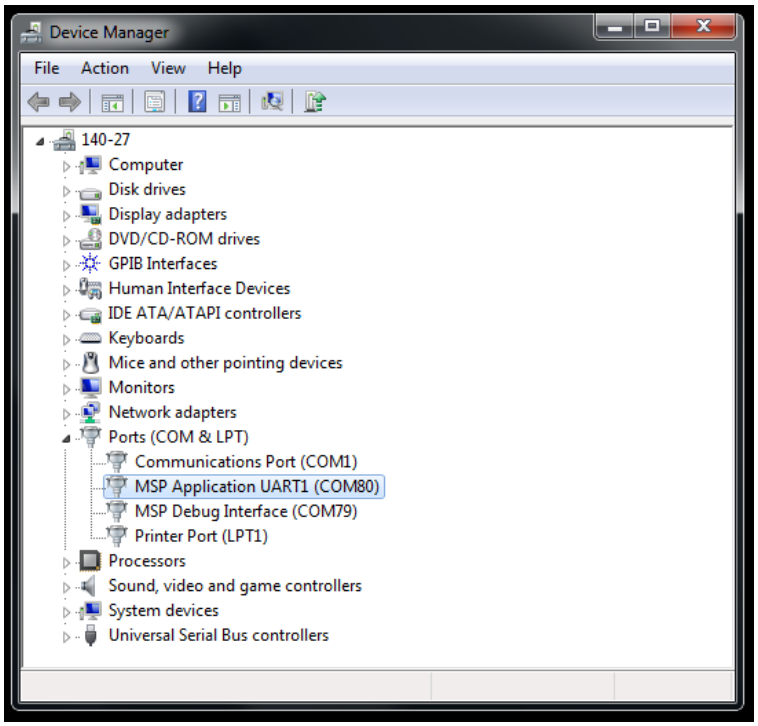


The Launchpad interfaces with your PC through a USB cable and you can program the microcontroller using a few different programs. In this class we will be using Energia, a software built by TI to look like the Arduino programming environment.

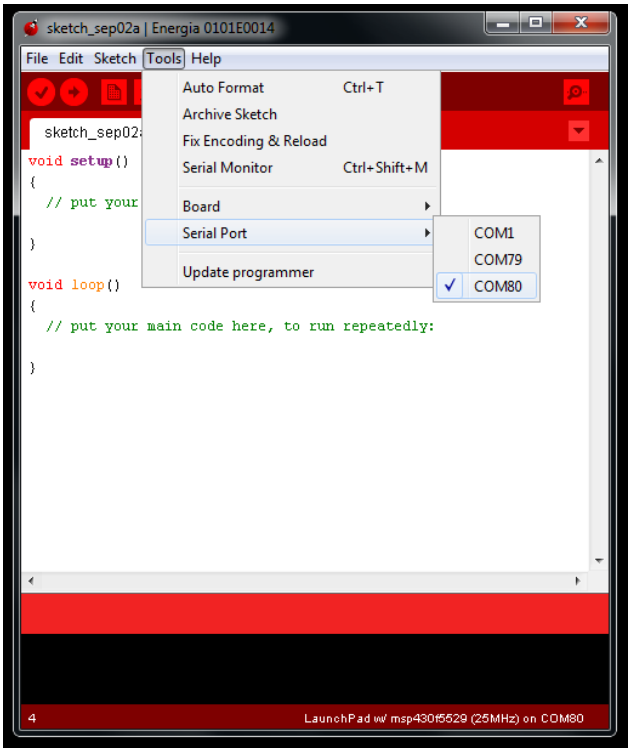
Let's start with loading some code on the Launchpad. Launch the Energia software and choose "Launchpad w/ `MSP430f5529 (25MHz)`" under `Tools > Board`. You need to make sure to choose one of the `MSP430f5529`'s when working with these boards.



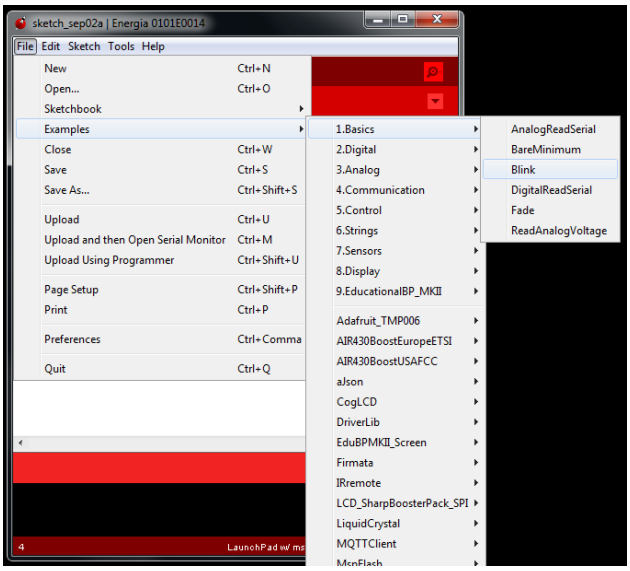
When you connect the Launchpad to the PC using the USB cable, the Launchpad starts 2 COM ports to communicate with the PC. One of these is used for the Application UART (Universal Asynchronous Receiver/Transmitter) while the other is used for the debug interface. To choose the right COM port to connect to, click the Windows Home button and type in "Device Manager". (If the search returns nothing, you will need to find Device Manager in the control panel). Then, click "Ports (COM LPT)". Note the COM value of the Application UART. **You need to do this every time you connect a Launchpad to a PC because the COM port might change.**



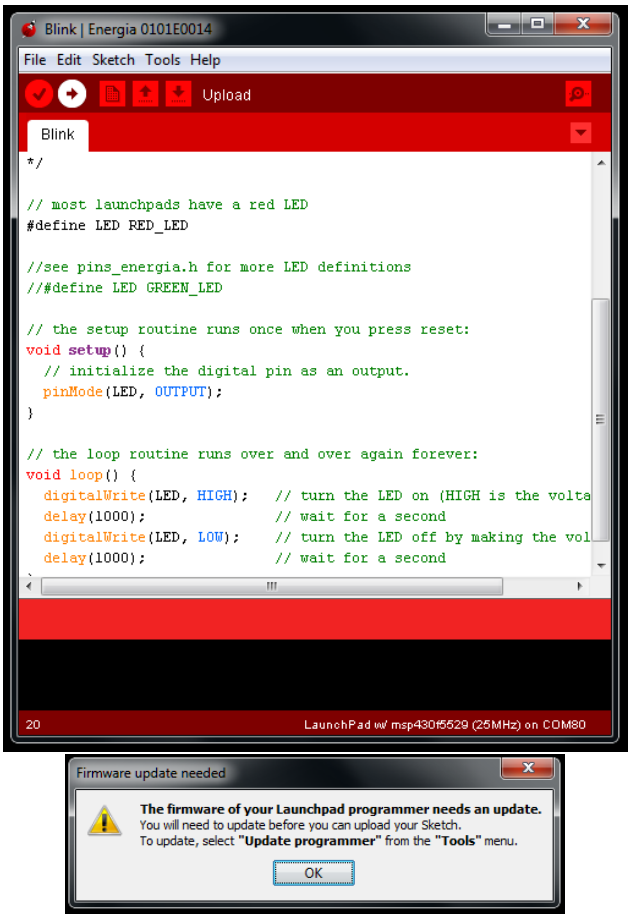
Under `Tools > Serial Port` choose the COM port you have noted before.



Now click `File > Examples > Basics > Blink`. This opens an example sketch (code) that blinks one of the LEDs on the Launchpad.



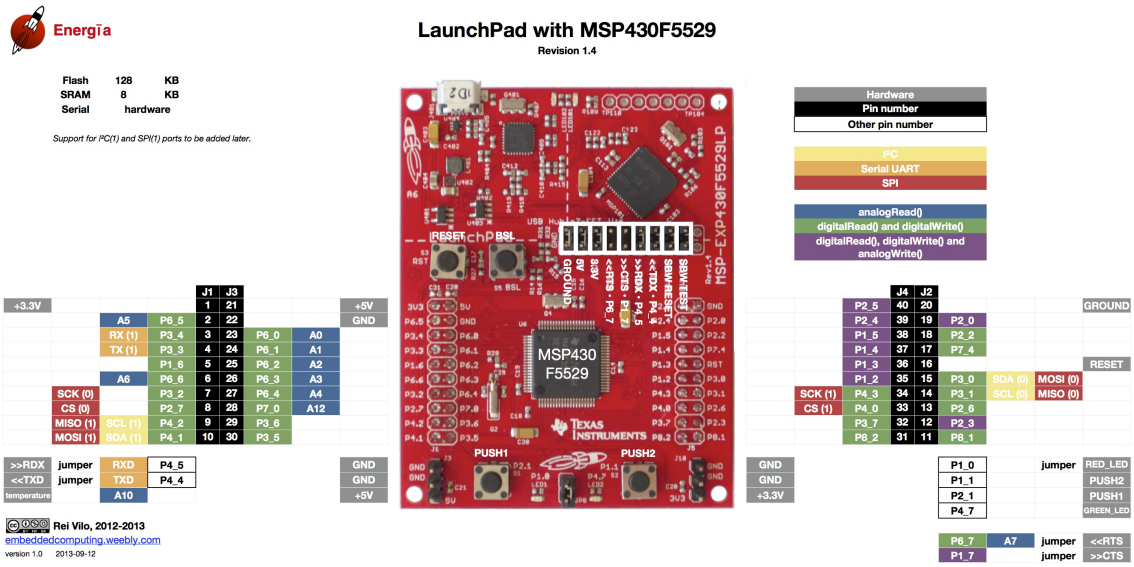
Click the Upload button (the right arrow in the top left of the window). If Energia asks for a firmware update, follow the instruction by clicking `Tools > Update Programmer`. Then, click on the upload button again.



Press the reset button (RST) in the Launchpad. The red LED on your Launchpad should blink. If it doesn't, get a GSI or lab helper to help you debug. **In general, always hit the reset button after every upload.**

There are multiple names that refer to the same pin in the Launchpad. For example, in the diagram below, 23 is the same as

P6_0


Energia
LaunchPad with MSP430F5529
Revision 1.4
Flash 128 KB
SRAM 8 KB
Serial hardware
Support for I²C(1) and SPI(1) ports to be added later.
MSP430 F5529
MSP-EXP430F5529LP
PUSH1
PUSH2
Hardware
Pin number
Other pin number
I²C
Serial UART
SPI
analogRead()
digitalRead() and digitalWrite()
digitalRead(), digitalWrite() and analogWrite()
Rei Vilo, 2012-2013
embeddedcomputing.weebly.com
version 1.0 2013-09-12