Successive Approximation ADC

By Josiah, Evan, and Sean

What is an analog vs digital reading?

- Analog signal is a continuous signal which represents physical measurements.
    - Examples: Temperature, pressure, and intensity
- Digital signals are discrete time signals generated by digital modulation.
    - Examples: Computer, CD’s, and watches
- In this class we are using Arduino Mega 2560 to interpret our analog signals and convert them into usable digital signals for recording data.

Everyday Examples of Analog to Digital

- When you talk on the cell-phone, analog voice into digital played on receivers phone
- Air conditioning units that take temperature readings and use that information to regulate room temperature.
- Control stick on gaming controllers
- Light sensors that measure how intense a beam of light is

Shortcomings

- Many errors can occur due to the nature of analog to digital converters that include but are not limited to:
    - **Temperature drift:** "is the behavior where for the same given physical quantity being measured, the sense element output is different at different temperatures. If this change in output due to temperature is not adequately compensated, it will appear as if the physical quantity being measured is changing." Example would be measuring Pressure
    - **Load Regulation:** As you draw more and more current into other devices(LCD, BME,etc), you can run into an error where this can cause your reference voltage of the ADC to start to experience an error. Can make your readings appear funny.

Sensors in 371 (Digital vs Analog)

Most components we work with transmit data using digital signals with I2C or SPI. However, a few of the simple sensors do not. These sensors have an analog voltage out pin. The voltage on this pin continually varies according to what the component is measuring. For the sensors in this course, this is generally a linear relation.

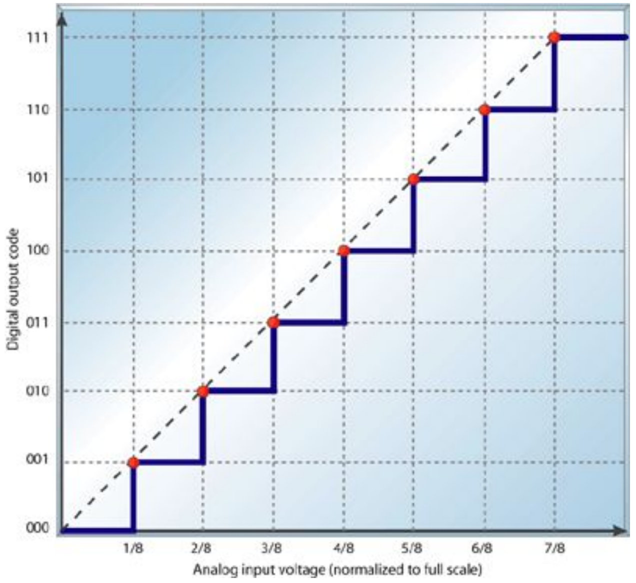
Digital	**Analog**
BME680 LSM9DS1 Accelerometer/Mag/Gyro MLX90614 IR Sensor	TMP36 Temperature Sensor Anemometer Wind Speed Sensor Electret Microphone w/ Amp

Analog to Digital Converters

Microcontrollers can only directly interpret digital signals. So they rely on Analog to Digital Converters (ADCs) to process analog signals. ADCs break the possible voltage range of analog signals into a finite levels. The nearest level can easily be represented digitally and passed on to the Microcontroller.

The actual process of finding the nearest level is complicated, and can be done in a few different ways.

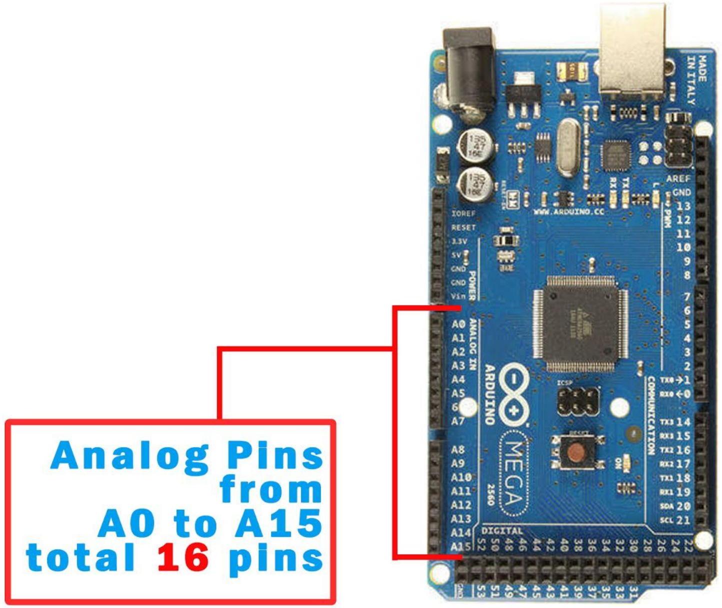
Direct-conversion, Successive-approximation, Ramp-compare, Integrating, and many more….



Arduino Mega 2560

The Arduino Mega 2560 has 16 total analog input pins. They are read by just one Successive Approximation ADC. The single ADC switches between pins with a Multiplexor. It has the following specs:

- Measures 0 - ~5 V
- 1024 Discrete Output Levels
- ~5 mV Resolution
- ~15k Samples / Second
- Slower With Switching
- Faster With Lower Resolution



Operation Modes

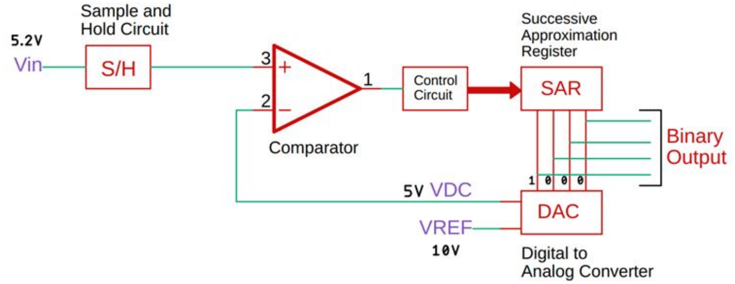
The Arduino Mega 2560’s ADC has three modes. It operates in single conversion mode by default. That mode is acceptable for nearly all cases. The other modes are significantly more complicated to set up.

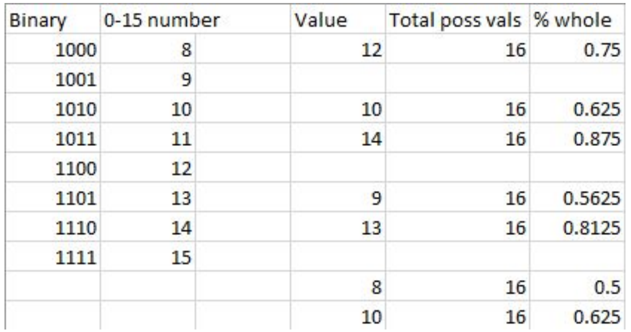
1. **Single Conversion** - The ADC make one conversion each time the Microprocessor code tells it.
2. **Triggered Conversion** - The ADC will make a conversion whenever a specified signal increases.
3. **Free Running** - The ADC will make a conversion whenever it can.

Successive Approximation Algorithm

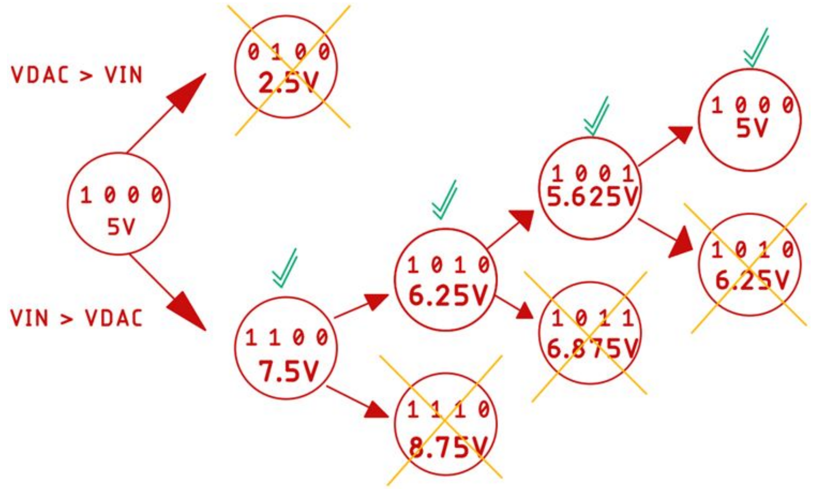
- Uses binary search by changing bits (e.g. 1111 to 0000)
    - Slow (sample rate < CPU clock rate)
- V_in is sampled and compared with a comparator logic circuit (AND, OR, etc.) to V_out of a DAC (Digital to Analog Converter)
- Comparator outputs comparison to SAR (Successive Approx. Register)
- SAR changes bit output based on comparator readings (increases or decreases the amount of 1's in the register)

Successive Approximation Algorithm (Cont.)





Binary	0-15 number		Value	Total poss vals	% whole
1000	8		12	16	0.75
1001	9				
1010	10		10	16	0.625
1011	11		14	16	0.875
1100	12				
1101	13		9	16	0.5625
1110	14		13	16	0.8125
1111	15				
			8	16	0.5
			10	16	0.625



References

https://bennthomsen.wordpress.com/arduino/peripherals/analogue-input/

https://en.wikipedia.org/wiki/Successive-approximation_ADC

https://en.wikipedia.org/wiki/Analog-to-digital_converter

https://www.maximintegrated.com/en/design/technical-documents/tutorials/7/748.html

https://circuitdigest.com/article/how-does-successive-approximation-sar-adc-work-and-where-is-it-best-used