Analog/Digital Conversion

The real world is analog. Interfacing a microprocessor-based system to real-world devices often requires conversion between the microprocessor's digital representation of values to an analog representation. We will focus on conversions to and from analog voltages; converting from electrical signals to other signals is the domain of sensors (e.g., thermistors) and transducers (e.g., speakers).

- Analog input signals are converted to digital values using analog-to-digital converters (ADCs).
- Analog output signals based on digital values are generated using digital-to-analog converters (DACs).
- ADCs and DACs are commonly available as single-chip devices that can be easily interfaced to microprocessor busses.

Outline

- Common conversion concepts
- Digital-to-analog conversion circuits
- Analog-to-digital conversion circuits

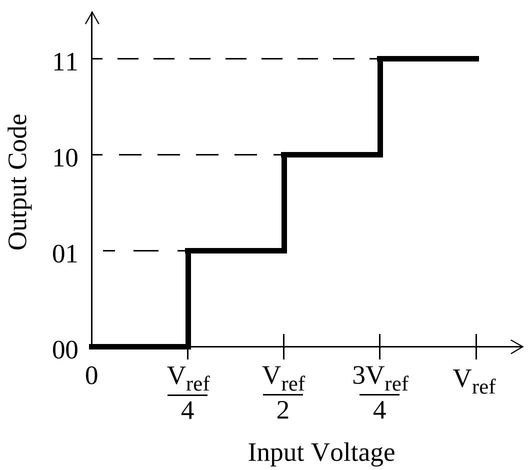
Basics

- The primary characteristic of a converter is its *resolution*, expressed as the number of significant data bits on the digital side of the converter. An *n*-bit converter divides an analog voltage range into 2^n sections, providing a resolution of 2^{-n} times the voltage range.

- *Error* is the difference between the analog voltage you believe a digital value represents and what that analog voltage acutally is. As we will see shortly, even an ideal converter introduces some error.

- *Accuracy* refers to how close an actual converter is to an ideal converter. Inaccuracies are another source of error.

- The graph below shows the transfer function for an ideal 2-bit ADC. The input voltage range (0, V_{ref}) is divided into 2^2 = 4 sections, so the ADC's resolution is 2^{-2} = 1/4 of V_{ref}.



Quantization Error and LSBs

- Each *code* (digital value) represents a range of analog inputs; e.g., the ADC will read '01' for any voltage in the range ($V_{ref}/4$, $V_{ref}/2$). The best we can do is assume that '01' means $3V_{ref}/8$. Since the actual voltage could be as low as $V_{ref}/4$ or as high as $V_{ref}/2$, there is a potential error of $\pm V_{ref}/8$. This error is called *quantization error*.

- Quantization error is inherent in the process of converting a continuous analog voltage to a finite number of discrete digital values. Even an ideal converter introduces quantization error.

- The absolute value of the quantization error in volts (along with most other types of conversion errors) depends on the voltage range (i.e. the value of V_{ref}) and the resolution of the converter. To normalize these parameters away, errors are typically expressed in terms of the ideal analog voltage difference represented by a unit change in the digital value.

- Since this unit change represents a change in the least significant bit of the digital value, this voltage difference is referred to as an *LSB.*

- Quantization error is always ±1/2 LSB.

Accuracy

- *Non-linearity* (or *absolute accuracy*) is the absolute deviation from the ideal transfer curve. The total error bound is the sum of the magnitudes of the absolute accuracy and the quantization error.

- *Differential non-linearity* is the deviation of the difference between two consecutive codes from the ideal 1 LSB difference. An absolute non-linearity of ±1/4 LSB could result in a differential non-linearity of ±1/2 LSB. The manufacturer may or may not specify a tighter bound on differential non-linearity.

- A converter is *monotonic* if an increase/decrease in the digital code always corresponds to an increase/decrease in the analog voltage. A non-monotonic converter by definition has > ±1/2 LSB non-linearity.

- *Full-scale error* (also called just *scale error*) is the deviation from the ideal at full scale (i.e. code is all 1's). Note that the ideal full scale is $(2^n-1)/2^n * V_{ref}$, not V_{ref}. Typically full-scale error (and its counterpart *zero error*) can be adjusted to 0 using external potentiometers, if necessary.

Conversion Time

- *Conversion time* is simply the time required to convert an input to an output. Depending on the type of converter (i.e., the internal design), conversion time can range from a few nanoseconds to a few milliseconds.

- As we will see shortly, designing converters is a three-way tradeoff between cost, conversion time, and accuracy.

- Some converters are internally pipelined to provide conversion rate > 1/(conversion time).

ADCs:

- Most ADCs provide an "end of conversion" signal that can be used as an interrupt input.

- A *sample-and-hold* ADC samples the analog input at the start of its conversion process and produces a code representing that specific voltage. An *averaging* ADC produces a code representing the average input voltage over the conversion time. Other ADCs may rely on you to not change the voltage (e.g. with an external sample-and-hold).

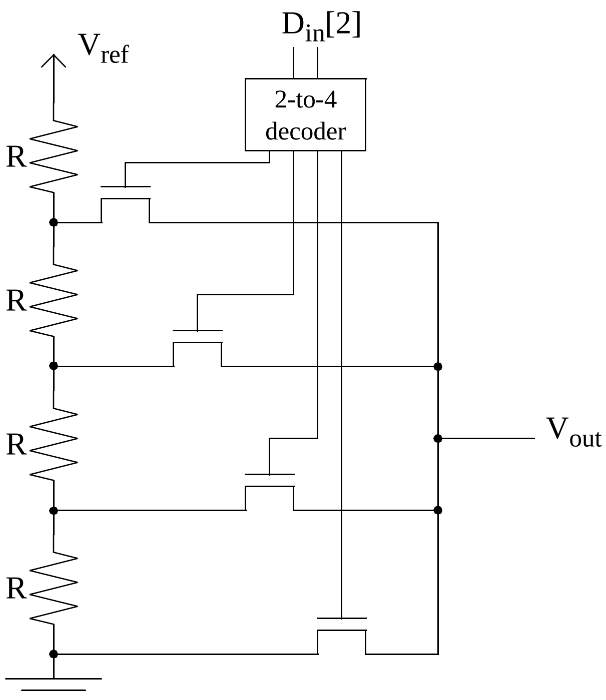
DACs:

- DAC conversion time is typically specified as the *settling time* required for the output to reach the specified accuracy.

- Most DACs can be driven faster than the specified conversion rate at a corresponding loss of accuracy.

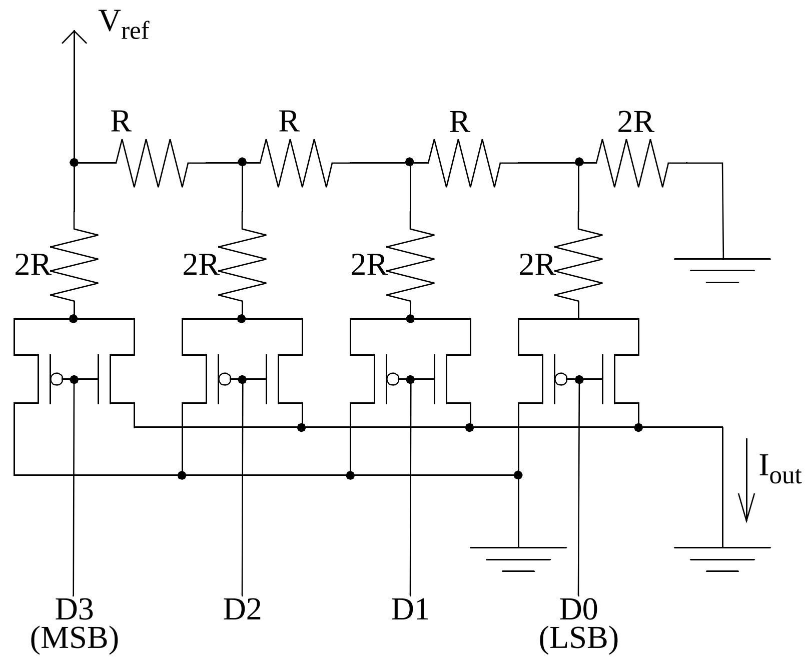
DAC Types

Voltage divider



- Fast
- Expensive: requires 2^n resistors, switches
- accuracy depends on matching all resistor values (but not exact resistor values)
- Guaranteed monotonic

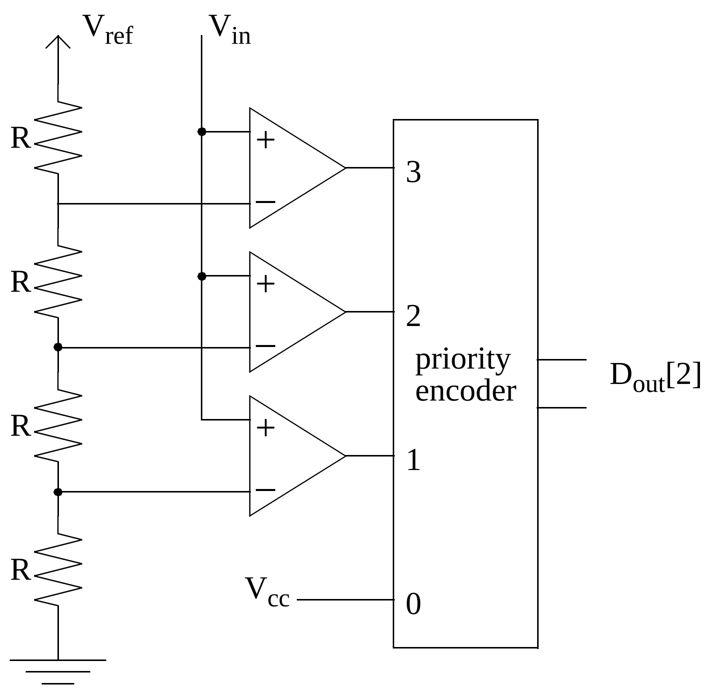
R/2R Ladder



- Cheaper: ~2*n* resistors, *n* switches
- Again, accuracy depends on matching all resistor values (but not exact resistor values)
- Harder to enforce monotonicity (consider 0111 -> 1000)
- Provides *current* output; op-amp required to convert to voltage, increases conversion time

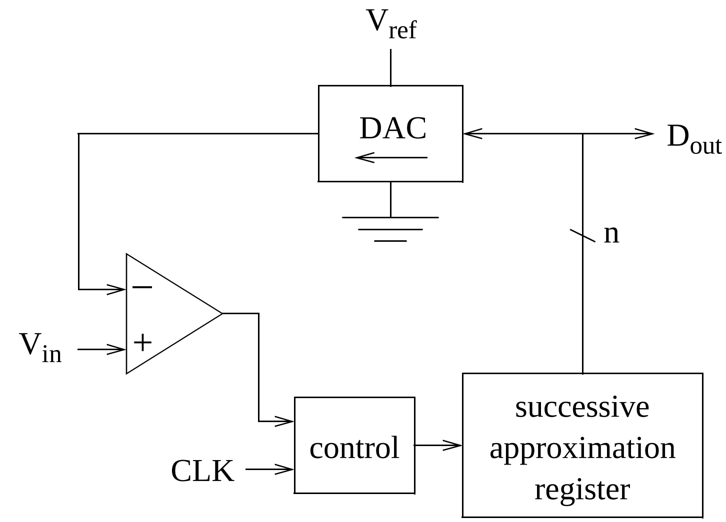
ADC Types

Flash



- ADC equivalent of voltage-divider DAC
- Same issues: fast but expensive (2^n resistors, 2^n-1 comparators)

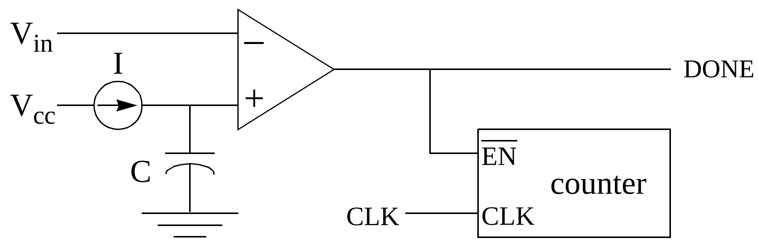
Successive Approximation (SA)



- binary search to match voltage
- Algorithm:
  1. Set successive approximation register to 0
  2. Starting at MSB, flip one bit to 1
  3. If DAC output < V_{in}, leave, else reset to 0
  4. go to next bit
- example: 4-bit ADC, Vref = 4.8V, Vin = 3.2V
- need fairly stable input through conversion process
- much cheaper than flash (only one comparator, 2^n or $2n$ resistors depending on DAC type)
- conversion time > *n* times DAC settling time

ADCs Cont'd

Single-slope Integration



- start: reset counter, discharge C
- charge C at fixed current I until $V_C > V_{in}$
- final counter value is D_{out}
- slow (can be many milliseconds)
- high resolution, good differential linearity
- absolute accuracy (linearity) depends on precision of C, clock, current source (I), etc.

Dual-slope Integration

- similar to single-slope, but uses full charge/discharge cycle to cancel out dependence on component values
- charge C from $I \propto Vin$ while counting from 0 to Dmax
- then discharge C at constant current while counting from 0
- final counter value is D_{out}
- eliminates dependence on precision of (most) components, including C, clock
- automatically compensates for compnent drift due to temperature, etc.
- inherently averaging
- very accurate (>20 bits)
- still slow