Physics 310
Lecture 9a – DAC and ADC

Mon. 3/19	**Ch 14.1, .6-.10; pp 373-374** (Sampling Frequency); **12.6:** ADC & DAC	**Project Proposal**
Wed. 3/21	More of the same	
Thurs. 3/22	**Lab 9:** ADC & DAC	**HW 9: A*** & Ch 14 Pr 13*, 17*
Fri. 3/23	More of the same; **Quiz** Ch 14	**Lab 9 Notebook**
Mon. 3/26	**Project:** Component Shopping	**Project Progress Report** (due at the beginning of
Wed. 3/28	Review *Sr. Seminar Presentations(quiz points)*	class)
Thurs. 3/29	**Exam 2**	

Announcements:

- **Project Proposals due Today**

Study List for Quiz #9:

- Analog-to-digital conversion (ADC)
  - How analog input and digital output are related
  - Conversion times
  - Nyquist criterion – sampling rate
- Digital-to-analog conversion (DAC)

Equation List:

Nyquist criterion: $f_{sample} > 2f_{max}$

Handout:
- Lab #9
- DAC circuits
- Flash ADC (2 bit)
- Project order list
-

14.1 Intro:
Remember the very first reading for this class, it was about the array of measurement devices and actuators – things that translate a physical property like a length, a time, a temperature, a pressure,… into a voltage, or translate a voltage into a force, a displacement,…All of these devices deal with inherently continuous properties and so produce or respond to a continuum of voltages. Analog circuitry is called for.
On the other hand, I don't have to sell you on the power of computers and other digital devices when it come to analyzing information, and these are inherently digital. So, if you want to bring the power of a computer to bear on a measurement or manipulation task, then you need to either convert the Analog information to Digital information or vice versa. That's the use of DACs (Digital to Analog Converters) and ADC's (Analog to Digital Converters).

The main point of today's reading is that these things can be done, but you generally won't design a circuit to do them – instead you'll use IC's. Still, it's important to know basically how they work, so you use them correctly / choose the right ones for your applications.

You might imagine that ADC's and DAC's are pretty mysterious and sophisticated things. In actuality, most operate on pretty simple principles. For digital to analog – essentially an adder where each bit line's value controls a gate that does/does not add in the corresponding voltage.

For analog to digital – essentially comparators are used to determine whether each bit should be on or off .

Digital (Binary) Background from Ch 11

While it was in the reading for Ch 11, we didn't really discuss binary, and an appreciation of it is necessary for an appreciation of what follows. So we'll pause to catch up on that.
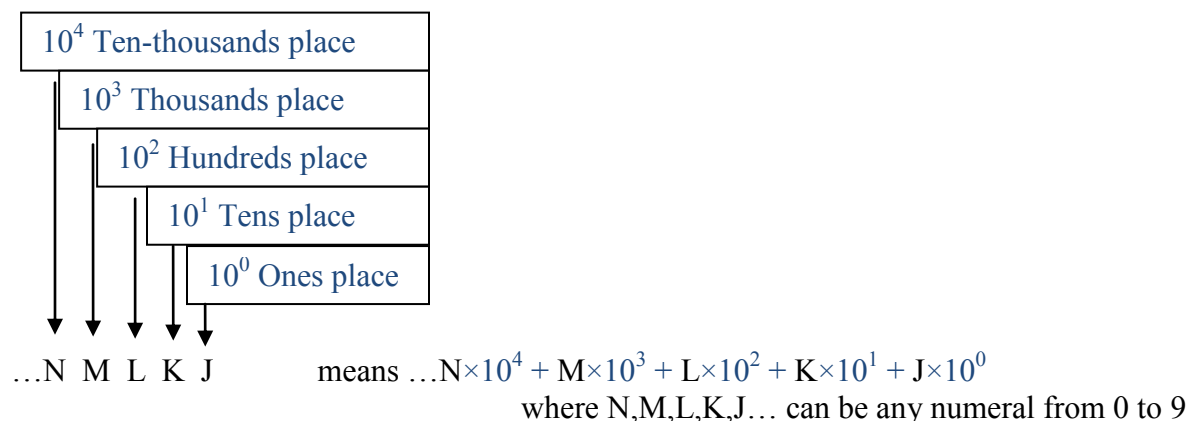
11-5 Numbering Systems

It's quite obvious that our old continuous-math circuits could represent and manipulate a continuous range of numeric values: the signal can be a voltage of 10.01V or 10.0099V or… That's particularly important for smooth operations like integration and differentiation. There are downsides though; for example, adders and integrators can't distinguish between signal and noise – they add / integrate *whatever* their input is. Along those lines, if you want to represent number to seven digits, you need to have a signal-to-noise ratio of 10^7!

Now, it's quite obvious that our new discrete-math circuits can represent and manipulate 0's and 1's, but what may not be immediately obvious is that, with 1's and 0's they can represent any discrete number to however many digits you're willing to handle; the catch is that rather than being able to encode the value in one continuously varying signal, you encode it in N discretely varying signals. We just have to represent that value in Binary.

Decimal a.k.a. Base 10

First let's quickly look at how we represent numbers in Decimal, then we'll see that it's not so very different in Binary. The two freedoms we have for representing different numbers are *symbols* and *placement* of those symbols. First we step through our basis of ten available symbols, 0-9, and then, if that's not enough, we add a 'place' and do it again.
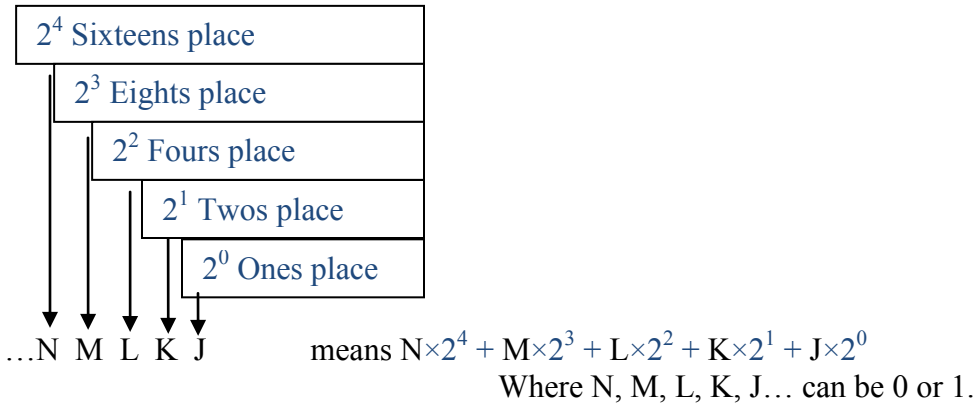
So, when we write 12,425 we've encoded the numeric value in the specific placement of specific symbols

10^4 Ten-thousands place

10^3 Thousands place

10^2 Hundreds place

10^1 Tens place

10^0 Ones place

…N M L K J   means …$N \times 10^4 + M \times 10^3 + L \times 10^2 + K \times 10^1 + J \times 10^0$

where N,M,L,K,J… can be any numeral from 0 to 9

Binary a.k.a. Base 2

Again, we've got the freedoms of symbols and placement with which to encode a number. Now, in Binary we only have a basis of two symbols, 0 and 1. So rather than having to move up to the next place after every 10th symbol, we have to do it after every 2nd symbol.

2^4 Sixteens place

2^3 Eights place

2^2 Fours place

2^1 Twos place

2^0 Ones place

$\ldots$N M L K J $\qquad$ means $N\times2^4 + M\times2^3 + L\times2^2 + K\times2^1 + J\times2^0$
$\qquad\qquad\qquad\qquad\qquad$ Where N, M, L, K, J… can be 0 or 1.

So, for example $10011 = 1\times2^4 + 0\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0 = 16 + 0 + 0 + 2 + 1 = 19$

Vocabulary

$\qquad$ Bit = one binary digit (which means one electronic 'line' carrying one signal)
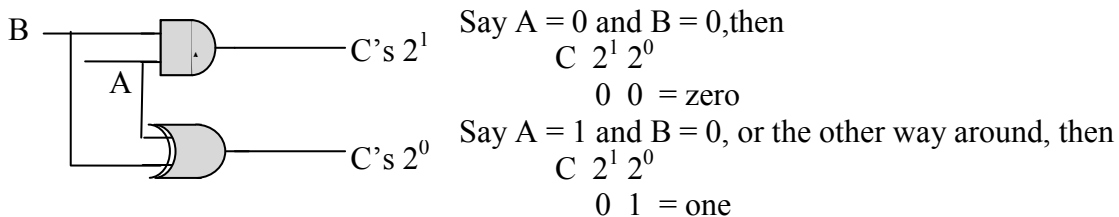$\qquad$ Byte = eight binary digits (which means eight electronic 'lines', each carrying one signal)
$\qquad\qquad$ With eight places, largest value is $11111111 =$
$1\times2^7+1\times2^6+1\times2^5+1\times2^4+1\times2^3+1\times2^2 +1\times2^1+1\times2^0 = 128+64+32+16+8+4+2+1=255$
$\qquad\qquad$ and of course the smallest is 00000000
$\qquad$ From 0 up to 255, there are $256=2^8$ discrete possible values. You may have encountered this language "8 bit", "16 bit", "32bit" and "64bit" when people talk about computers. For example, you can set your monitor to 16bit or 32bit color: $2^{16}=65,536$ shades or $2^{32}=4,294,967,296$ shades. 64bit logic means that a value can be represented across 64 discrete lines.

We'll get more experience with this later, but just to relate this 'binary' stuff back to our logic gates, here's a really simple operation: adding two one-bit numbers and encoding that as a two-bit number:
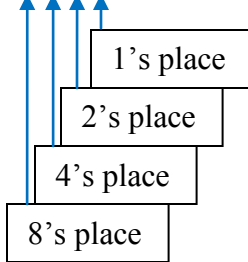
B

A

C's 2^1

C's 2^0

Say A = 0 and B = 0,then
$\qquad$ C $\quad 2^1 \; 2^0$
$\qquad\qquad$ 0 0 = zero
Say A = 1 and B = 0, or the other way around, then
$\qquad$ C $\quad 2^1 \; 2^0$
$\qquad\qquad$ 0 1 = one

Say A = 1 and B= 1
C 2^1 2^0
1 0 = two

14.6 Digital –to-Analog Converters: DACs
Weighted Current Source DAC
Here's a very simple DAC. Say you have a 4-bit signal that's supposed to represent a numeric value in binary with four lines: DCBA.
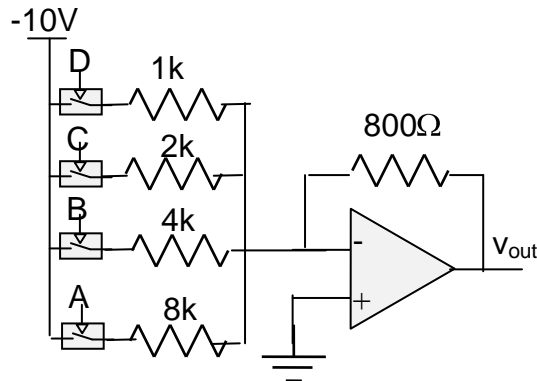
| 1's place |
| 2's place |
| 4's place |
| 8's place |

Conceptually, when we translate from a binary representation to a decimal representation, of, say
DCBA
1 0 0 1
That would represent 9.
We add 1 + 8 (because those are the two lines that are Hi) to get 9.
Then the simple circuitry analog is to have an adder that, using some switches, decides whether or not to add 1V to 2V to 4V to 8V:
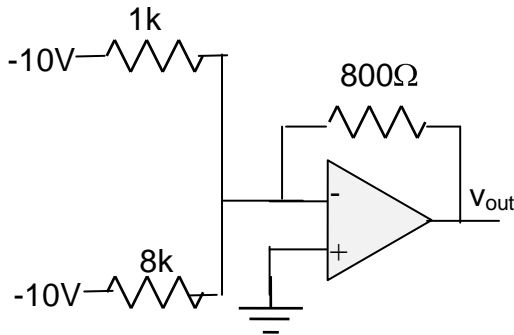

The little boxes represent transistor switches. For example, when A is Hi, the bottom switch is connected. So, if we have
DCBA
1 0 0 1
Then the circuit is essentially

So,

$$V_{out} = -\left(V_{in.1}\frac{R_f}{R_1} + V_{in.2}\frac{R_f}{R_2} + \ldots\right)$$

$$= -\left(-10V\frac{800}{1000} - 10V\frac{800}{8000}\right)$$

$$= -8V - 1V = 9V$$

Practical Disadvantage: A wide range of resistors must be balanced and a wide range of currents pass through the switches which are, in practice, transistors. The current through the switch for the least significant bit (line A in this example) may be fairly large and thus the capacitance across that switch could be fairly large which means the switch could be *slow*. Another issue is that resistors of different values have to be balanced just right, and that's hard to achieve.

A related approach would be using current sources instead of resistors. Of course, that's in effect what we're doing in the example above.
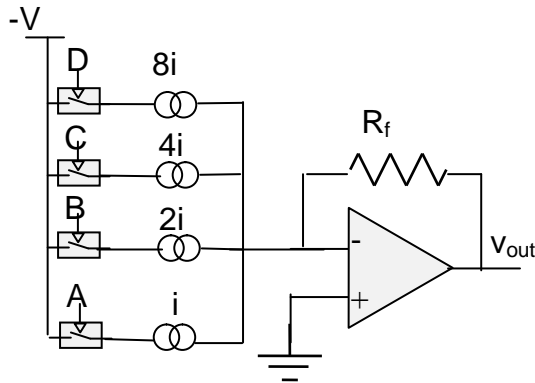
$V_{out} = R_f i_f$ where i_f is the sum of whichever currents are switched in. Note that my version has a negative voltage supply; that's so V_{out} will be positive. Alternatively, if the voltage supply is positive, V_{out} will be negative.

The book gives a practical realization of a similar circuit (but with a positive voltage line) using transistors and diodes for the switches. One of the switched current sources is essentially

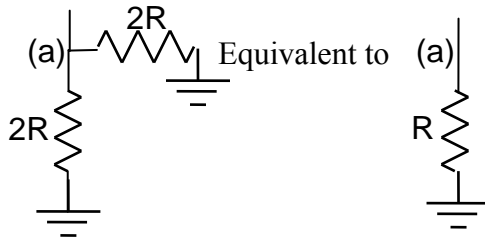Equivalent to
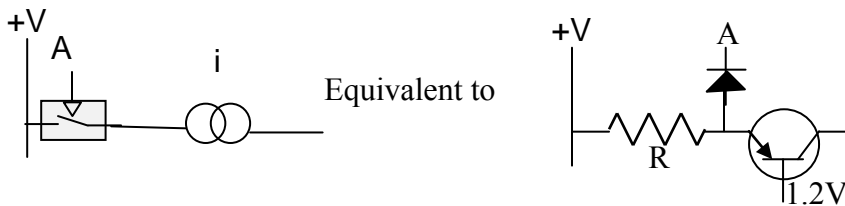
R-2R Ladder Method DAC

Here's a variation on the theme we're working with; it addresses one of the key issues – the difficulty of balancing umpteen different resistors. This circuit manages to use just two different resistor values: R and 2R. Note: I've tried to make the geometry more similar to that which we've already seen in our adders, but the logic is the same as in the book's version.
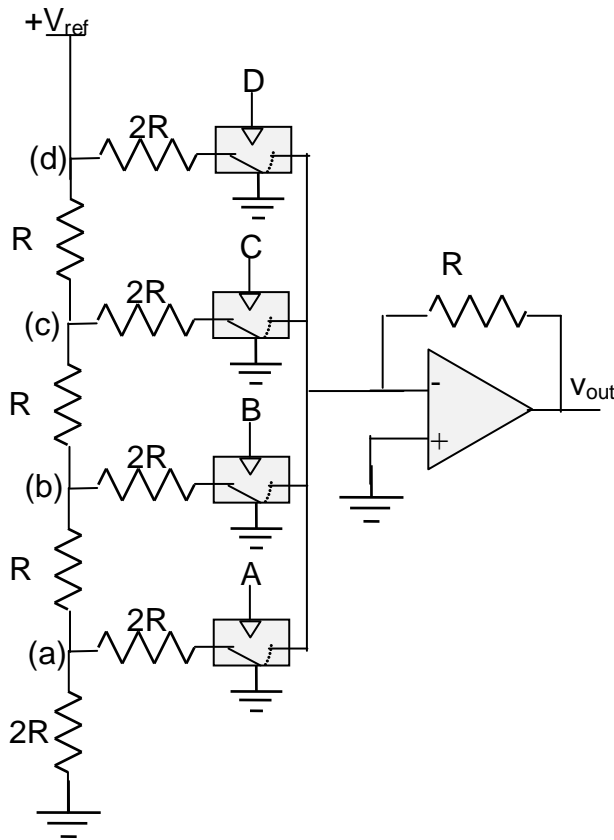
So, how does this work? (Following Faissler p. 308-309)
1) Regardless of the position of the switch, the right end of each 2R connects to ground (or a 'virtual' ground thanks to the op-amp.) So the amount of current passing through any one of these resistors is insensitive to the position of any of the switches.
2) Just consider the bottom branches, below and to the right of point (a). These two 2R resistor are in parallel to each other, both connect (a) to ground. So, from the perspective of point (a), there might as well just be a single R resistor to ground.

But if that's true, then from point (b)'s perspective the circuit below and to the right of it is equivalent to
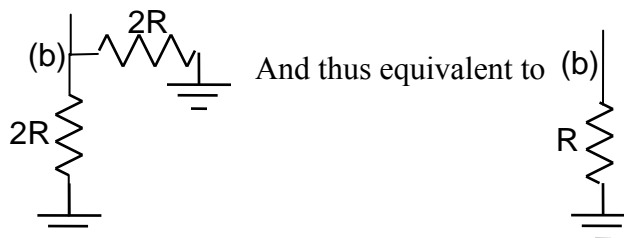
Ditto for points (c) and (d) and, should we extend this circuit by adding more and more branches, it would be true for their points (e), (f),…

The moral is that each of these junction points sees itself as being just 1R above ground when it looks downstream. Now, what does each point see when it looks *up* stream?

3) Now, point (d)'s voltage is clearly $V_d=V_{ref}$.
   a. Then, from the perspective of point (c), it's in the middle of a voltage divider so
   $V_c= \frac{1}{2} V_d = \frac{1}{2} V_{ref}$
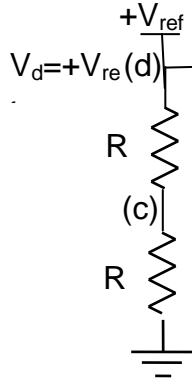
   b. Similarly, from the perspective of point (b), it's in the middle of a voltage divider so $V_b= \frac{1}{2} V_c = \frac{1}{2}\frac{1}{2} V_{ref}= \frac{1}{4} V_{ref}$.
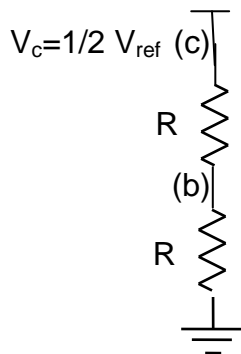

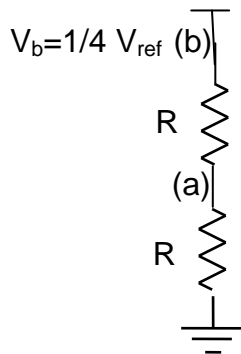   c. And then, of course, from the perspective of point (a), it's in the middle of a voltage divider so $V_a= \frac{1}{2} V_b =\frac{1}{2} \frac{1}{4} V_{ref} = \frac{1}{8} V_{ref}$.

Okay, labeling these points with their voltage values, we can see that we have something fairly simple.

The output is then just that of an adder with the appropriate input voltages switched in.

$$V_{out} = -\left(V_{ref} \frac{R}{2R}(D) + \frac{1}{2}V_{ref}\frac{R}{2R}(C) + \frac{1}{4}V_{ref}\frac{R}{2R}(B) + \frac{1}{8}V_{ref}\frac{R}{2R}(A) \right)$$

$$V_{out} = -\frac{V_{ref}}{2}\left(D + \frac{1}{2}C + \frac{1}{4}B + \frac{1}{8}A \right)$$

> Where A, B, C, and D are either 1 or 0, depending on whether their switches make the connections. If $V_{ref} = -16V$, then V_{out} has the value equivalent to the binary number represented by DCBA.

> **$V_{out.max.}$** For future reference, notice what the maximum output value is:

$$V_{out.max} = -\frac{V_{ref}}{2}\left(1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} \right) = -V_{ref}\frac{1+2+4+8}{16} = -V_{ref}\frac{15}{16} = -V_{ref}\frac{2^4-1}{2^4}$$

> More generally, if we had n data lines / bits, the maximum output would be

$$\boxed{V_{out.max} = -V_{ref}\frac{2^n-1}{2^n}}$$

Multiplying and De-glitching DACs
The format of my final expression for V_{out} gives you a sense of why the R-2R Ladder is classified as a "multiplying" DAC – the output is the "product" of a reference voltage and the logic encoded in the digital signal.

Deglitching DACs are relatively expensive – they largely compensate for dips in the signal that occur when a switch is switched.

Summary: Commercial DAC's feature the necessary resistor networks, the "feedback" resistor, R_f, and sometimes the reference voltage source, but they do not generally contain the op-amp. For example,
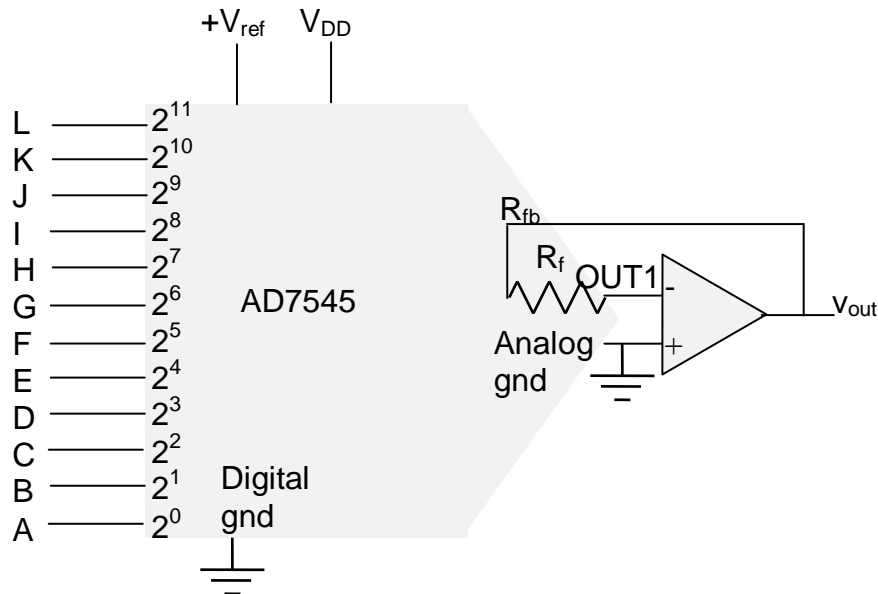
p.s. the "AD" in this part's name are the initials of the company that makes it, Analogue Devices, not a reference to the function it performs – that information is mysteriously conveyed by the "7545."

- **Conversion:** If a DAC has n digital lines, i.e. bits, coming in, then the largest value it can represent is with all n of them high. In simple binary, that corresponds to a value of 2^n-1. Where does the "-1" come from? Consider this example: we have 4 bits so the largest binary number is
    - 1 1 1 1
    - $2^0+2^1+2^2+2^3 = 1+2+4+8=15=16 -1=2^4-1$.
    - The same thing happens in decimal: what's the largest number that a 2-digit decimal display can represent? $99 = 100-1 = 10^2-1$.
  - Anyway, if this value would yield a full-scale output, then we've got a conversion factor of
    - $$V_{out} = Integer_{in} \frac{V_{out.\max}}{Integer_{in.\max}}$$ Now, as we saw for the ladder scheme,
    $$V_{out} = Integer_{in} \frac{V_{out.\max}}{2^n - 1}$$
    $$V_{out.\max} = V_{ref} \frac{2^n - 1}{2^n}$$ that means that
    - $$\boxed{V_{out} = Integer_{in} \frac{V_{ref}}{2^n}}$$

Two key qualities by which different DAC's are distinguished from each other are their resolution and their speed.

- **Resolution:** Often one says that the "resolution" of a DAC or of an ADC is the number of bits it handles, n. This largely determines how smoothly the output of a DAC can vary or how small a difference an ADC can resolve. Since the possible digital values are

integers, (1, 2, 3,…), obviously the steps in consecutive digital values are size 1. The corresponding steps in the analog output values are

- $$\Delta V_{out.\min} = \Delta Integer_{in.\min} \frac{V_{ref}}{2^n} = 1 \frac{V_{ref}}{2^n}$$

$$\boxed{\Delta V_{out.\min} = \frac{V_{ref}}{2^n}}$$

  o So that's the smallest difference this can 'resolve.' To put a value to that, a 12-bit DAC with a 10V reference could output values in steps of 2.4mV. For many purposes, that's as good as zero; it's as good as being able to output a continuum of values.

- **Settling Time:** This is the time it takes for an input value to propagate through to the intended output value. Generally, it's on the order of ns to μs.