5 Analog/Digital Conversion

In this section we discuss the important topic of analog to digital conversion (often written A/D), and digital to analog conversion (D/A). On one hand, most electrical measurements are intrinsically analog. To take advantage of the great capabilities available for digital data storage, processing, and computation, on the other hand, requires the conversion of analog to digital. Hence, analog to digital (A/D) conversion techniques have become extremely important. A great deal of technical effort has gone into producing A/D converters (ADCs) which are fast, accurate, and cheap. D/A converters (DACs) are also very important. For example, video monitors convert digital information generated by computers to analog signals which are used to direct the electron beam at a specified portion of the monitor screen. DACs are conceptually simpler than ADCs, although it is diffcult in practice to build a precise DAC.

We will discuss D/A conversion before A/D. But first we go over some underlying ideas.

5.1 A/D Resolution

First of all we should keep in mind that there are several different schemes for encoding analog information as bits, depending upon what is required by a particular application. One extreme is that of encoding the complete analog signal in as much detail as possible. For example, a musical instrument produces an analog signal which is readily converted to an analog electrical signal using a microphone. If this is to be recorded digitally, one naturally would choose to digitize enough information so that when the recording is played back, the resulting audio is not perceived to be significantly different from the original. In this case the analog signal is a voltage which varies with time, $V(t)$.

At any time t_0, $V(t_0)$ can be sampled and converted to digital. The analog signal must be sampled for a finite time, called the *sampling time*, Δt. One may guess that it is necessary to sample the analog signal continuously, with no gaps between consecutive samples. This turns out to be overkill. The *Nyquist Theorem* states that if the maximum frequency of inerest in the analog input is $f_{\max}$, then perfect reproduction only requires that the sampling frequency f_{samp} be slightly greater than twice $f_{\max}$. That is,

$$f_{\text{samp}} > 2f_{\max}$$

For example, for audio signals the maximum frequency of interest is usually 20 kHz. In this case the input analog must be sampled at a little over 40 kHz. In fact, 44 kHz is typically used.

Alternatively, it might not be of interest to represent the entire analog input digitally. Perhaps only one feature of the analog signal is useful. One example is "peak sensing," where one samples and digitizes the input only at the instant where an instrument's output achieves a maximum analog output. Or one may average ("integrate") an input signal over some predefined time, retaining only the average value to be digitized.

For any of these sampling schemes, there remains the issue of how many bits are to be used to describe the sampled signal $V(t_0)$. This is the question of A/D *resolution*. We need a standard definition of resolution. Let's say, for example, that we choose to digitize the input using 12 bits. This means that we will try to match our analog input to 1 of $2^{12} = 4096$ possible levels. This is generally done by ascribing a number from 0 to 4095. So, assuming our ADC works correctly, the digital estimate of the analog input can, at worst, be wrong by the range of the LSB. On average, the error is half of this. This defines the resolution. Therefore, for our 12-bit example, the resolution is $1/(2 \cdot 4096)$, or a little worse than 0.01%.

5.2 D/A Conversion

The basic element of a DAC is the simplest analog divider: the resistor. First, we need to review the two important properties of an operational amplifier ("op-amp") connected in the inverting configuration. This is shown in Fig. 25. The two important properties are

1. The "−" input is effectively at ground. ("virtual ground")

2. The voltage gain is $G \equiv V_{out}/V_{in} = -R_2/R_1$. An equivalent statement is that for a current at the − input of $I_{in} = V_{in}/R_1$, the output voltage is $V_{out} = GV_{in} = -R_2 I = -V_{in}R_2/R_1$. Sometimes this is written in the form $V_{out} = gI_{in}$, where g is the *transconductance*, and $g = -R_2$ in this case.

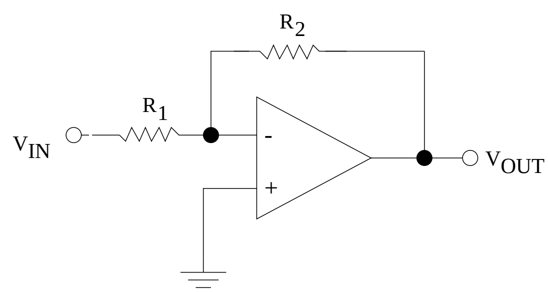

Figure 25: Inverting op-amp configuration.

The basic idea of most DACs is then made clear by the 4-bit example illustrated in Fig. 26. The input 4-bit digital signal defines the position of the switches labelled a_0–a_3. A `HIGH` input bit would correspond to a switch connected to 1.0 V, whereas a `LOW` connects to ground. The configuration in the figure represents a binary input of 1010, or 10_{10}. Since the virtual ground keeps the op-amp input at ground, then for a switch connected to ground, there can be no current flow. However, for switches connected to 1.0 V, the current presented to the op-amp will be 1.0 V divided by the resistance of that leg. All legs with `HIGH` switches then contribute some current. With the binary progression of resistance values shown in the figure, the desired result is obtained. So for the example shown, the total current to the op-amp is $I = 1.0/R + 1.0/(4R) = 5/(4R)$. The output voltage is

$$V_{out} = -RI = 5/4 = 1.25\text{V}$$

When all input bits are `HIGH` ($1111 = 15_{10}$), we find $V_{out} = 15/8$ V. A simple check of our scheme shows that

$$(5/4)/(15/8) = 2/3 = 10/15 = 1010/1111$$

as expected.

5.2.1 The R-2R Ladder

This represents a rather minor point, although it is an interesting idea. The "R-2R ladder" is of practical interest because it uses only two resistor values. Since it is difficult to accurately fabricate resistors of arbitrary resistance, this is beneficial. The two resistances of the R-2R

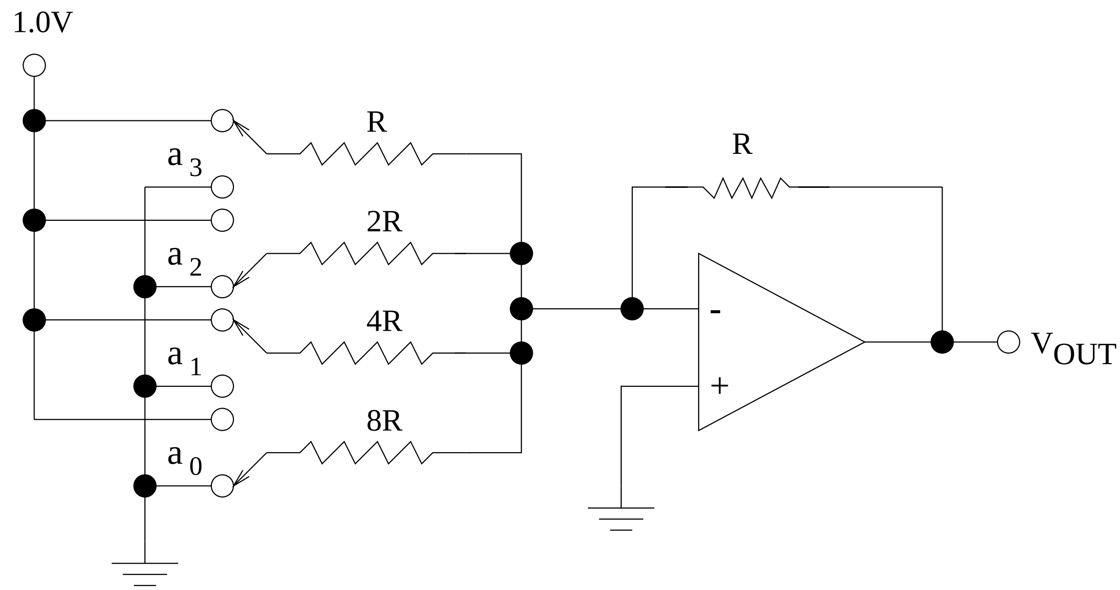

Figure 26: Example 4-bit DAC scheme.

are to be contrasted with the scheme represented by the circuit of Fig. 26, which employs as many resistance values as there are bits. The idea behind the R-2R ladder hinges on noticing the pattern of equivalences represented by Fig. 27, which can be used to replicate an arbitrarily long ladder, and hence handle in arbitrary number of bits.

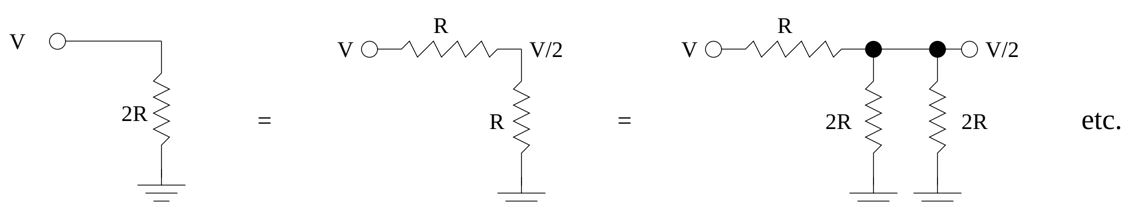

Figure 27: Principle of the R-2R ladder. The rightmost 2R resistor can be indefinitely relicated with this equivalent circuit.

5.3 A/D Conversion

ADCs fall into 3 general types of technique:

(1) parallel encoding (flash): fast; limited accuracy

(2) successive approx. (feedback): med. fast; good accuracy

(3) single or double slope: slow; best potential accuracy

All of these techniques use a device known as a *comparator*. This was discussed in 431/531 and in the text Chapters 4 and 9. Here, we will not discuss how comparators work, but we do need to know what they do. There are many makes of comparators. We will use the model LM311 in lab. Figure 28 shows a comparator schematically. Internally, the comparator can be thought of as a fast, very high-gain differential amplifier ("A") with inputs "+" and "−." We can put a "threshold voltage" at the "−" input. Call it $V_{\rm th}$. The circuit input $V_{\rm in}$ is connected to the "+" input. When $V_{\rm in} > V_{\rm th}$, the comparator amplifies this difference until the output reaches its largest possible value, which is determined by the connection through the pull-up resistor. In the configuration shown here, as well as in Lab 5, the ~ 1 kΩ pull-up resistor is connected to +5 V. (Note that while +5 V is convenient for many digital circuits, it is possible to use other values, such as +12 V.) When $V_{\rm in} < V_{\rm th}$, the output swings the other way. This level is usually determined by a connection to one of the comparator pins. Here, it is ground.

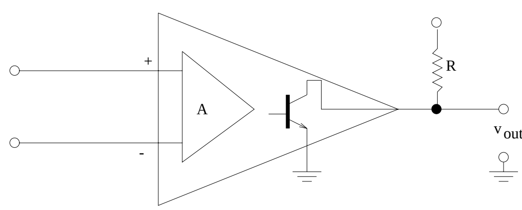

Figure 28: Comparator.

Hence, the comparator represents a one-bit ADC. When the analog input exceeds the pre-defined threshold, the output goes to digital `HIGH`, and when the input is less that the threshold, the output goes to digital `LOW`.

5.3.1 Flash ADCs

In this scheme, the input is fanned out in parallel to several comparators with monotonically increasing thresholds. The pattern of comparator outputs is then analyzed by some combinational logic (*i.e.* gates) to determine the output. This technique is called *flash* (or *parallel*) *encoding*. We exemplify the flash ADC scheme with the 2-bit ADC shown in Fig. 29. With $n = 2$ bits, we need to define $2^n = 4$ possible states. These states represent 4 separate intervals. The analog input will fall into one of these intervals, and we will encode this assignment with the 2 bits. Defining the boundaries of 2^n intervals requires $2^n - 1$ comparators, with the threshold of each comparator set to the appropriate boundary voltage.

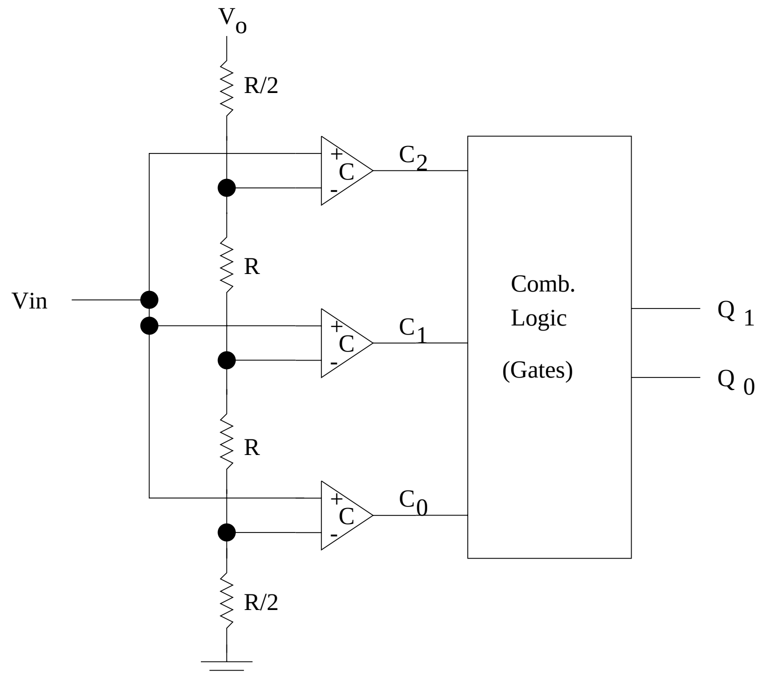

Figure 29: Schematic of a 2-bit flash ADC.

Let's go through a concrete example. Assume that our FADC circuit is designed to handle analog voltage input signals in the range -0.5 to 3.5 V. Thus, we have a 4-volt total input range, with each interval spanning 1.0 V. Therefore, each state will have a maximum error, or resolution, of half the interval, or 0.5 V. (This is $4.0/(2 \cdot 2^n)$, as we said previously in our definition of resolution.) So an input which is in the range 2.5–3.5 V will give a HIGH output only to comparator output C_2, and our digital estimation will correspond to 3.0 V. Hence, the threshold for the upper comparator (its "$-$" input) should be set at 2.5 V. Similarly for the remaining comparators we work out the values which are given in the table below, where V_{est} is the digital estimate which corresponds to each state.

V_{in} range	Comparator	Threshold	V_{est}	$C_2C_1C_0$	Q_1Q_0
2.5–3.5 V	C_2	2.5 V	3.0 V	111	11
1.5–2.5 V	C_1	1.5 V	2.0 V	110	10
0.5–1.5 V	C_0	0.5 V	1.0 V	100	01
-0.5–0.5 V	–	–	0.0 V	000	00

Using Ohm's and Kirchoff's Laws, we arrive at the resistance ratios shown in Fig. 29 in order to achieve the desired comparator thresholds. All that remains is to determine the gate logic to convert the pattern of comparator outputs to a 2-bit digital output. Generalizing from the above, we see that we have agreement with our previous statements: For an n-bit ADC, we require $2^n - 1$ comparators, and the resolution is $\Delta V/2^{n+1}$, where ΔV is the full range of analog input.

5.3.2 Successive Approximation ADCs

This technique is illustrated by Fig. 30, which is also the one given for Lab 5. It uses a digital feedback loop which iterates once on successive clock cycles. The function of the *successive approximation register*, or SAR, is to make a digital estimate of the analog input based on the 1-bit output of the comparator. The current SAR estimate is then converted back to analog by the DAC and compared with the input. The cycle repeats until the "best" estimate is achieved. When that occurs, this present best estimate is latched into the output register (written into memory). By far the most common algorithm employed by SARs is the *binary search algorithm.* This is the one used by the SAR in Lab 5, and is illustrated in the example in the next secion.

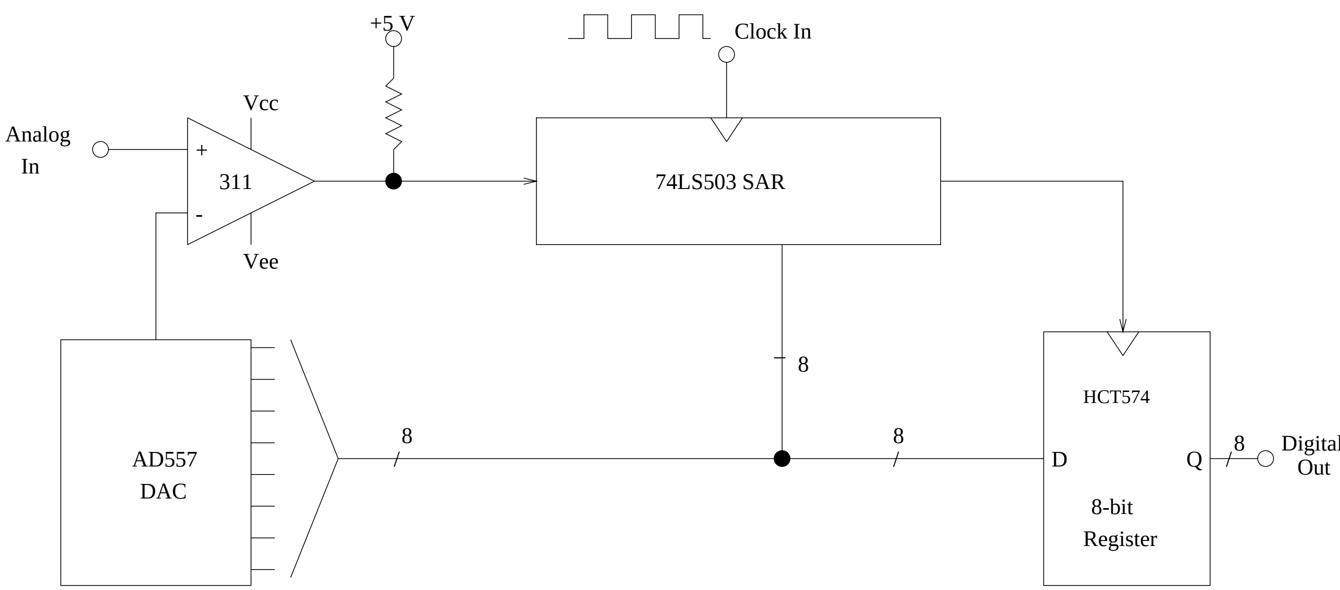

Figure 30: Scheme for 8-bit successive approximation, or feedback, ADC.

5.3.3 Binary Search Example

In this example we will see the binary search algorithm in action. The binary search algorithm can be summarized with the following words: *Go to the midpoint of the remaining non-excluded range.* In our example, we assume an 8-bit ADC with an expected input voltage range of 0 to 10 V. So, naturally we choose the digital output to be $00000000_2 = 0$ when the input is 0 V, and $11111111_2 = 255$ when the input is 10 V. Hence, the LSB represents a voltage step $\Delta V = 10/255 = 39.22$ mV.

Let the input voltage be some arbitrary value, 7.09 V. Now let's see how the algorithm works. Translating the words for the algorithm, written above, to what the SAR actually does is straightforward. The SAR always outputs one of two results, depending upon whether the output from the comparator was `TRUE` or `FALSE`. More precisely, the comparator will issue a `HIGH` if the current estimate is too small compared to the actual input, or a `LOW` if it is too big. The SAR then does the following:

1. If estimate too small, add 1 to MSB$-(n+1)$; or

2. If estimate too big, subtract 1 from MSB$-(n+1)$.

where n is the current clock cycle (see table).

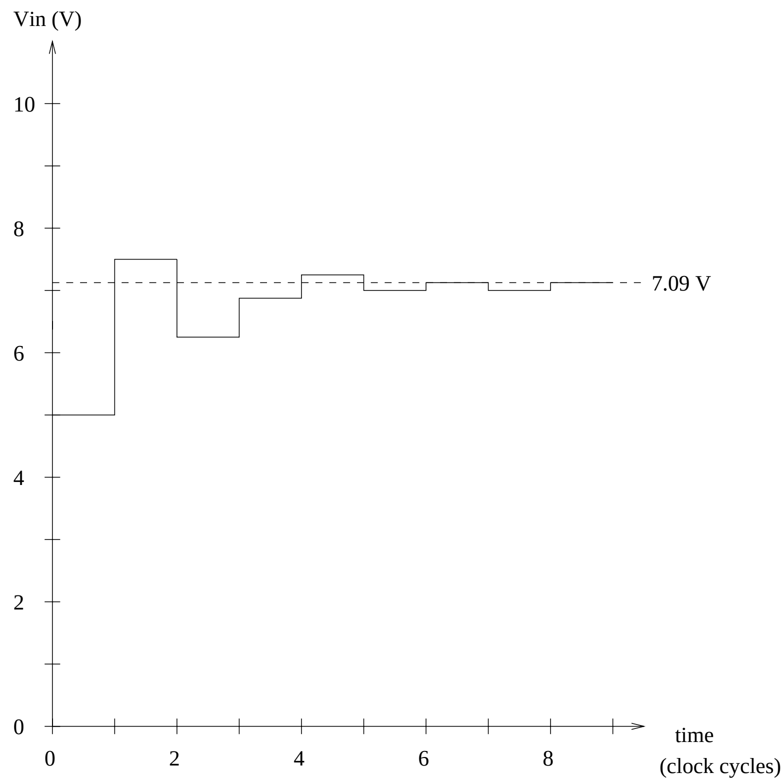

Figure 31: Binary search algorithm in action. The analog input is 7.09 V. The digital estimate for each clock cycle is represented by the solid line, and corresponds to the value of V_{est} in the table below.

Clock Cycle, n	SAR Bits	SAR Bit Sum	V_{est} (V)	comp. decision
0	01111111	127	4.98	too small
1	10111111	$127+64=191$	7.49	too big
2	10011111	$191-32=159$	6.24	too small
3	10101111	$159+16=175$	6.86	too small
4	10110111	$175+8=183$	7.18	too big
5	10110011	$183-4=179$	7.02	too small
6	10110101	$179+2=181$	7.10	too big
7	10110100	$181-1=180$	7.06	too small

The binary search algorithm is guaranteed to find the best possible estimate in a number of clock cycles equal to the number of bits. In the example above, the best estimate was actually determined on the seventh clock cycle ($n=6$). But since the input value was between the digital estimates 180 and 181, there was no way for the ADC to determine which estimate was closer to the actual input value (without adding one more bit). Since the input can fall anywhere within 180 and 181 with equal likelihood, there should be no bias introduced with this method due to systematically choosing a digital estimate which is too small or too big. This is the desired outcome.

The binary-search algorithm is fast and efficient, and also has the advantage that it completes its estimation in a well determined number of clock cycles. Hence, the final digital result can always be latched after n clock cycles, where n is the number of bits. (Many ADCs actually wait one additional clock cycle in order to guarantee that bits have settled, are latched properly, and are reset for the next input.

5.3.4 Single/Dual Slope ADCs

These techniques are slower than flash or successive approximation, but in principle can be quite accurate. The improved accuracy is for two reasons, because time, which is robustly measured using digital techniques, is used as the measured quantity, and because there is some immunity to noise pickup, especially for the dual slope case.

The single slope technique is illustrated in Fig. 32, which is taken from Figure 9.54 of the text. The device near the input and the capacitor is an FET transistor which is used as a switch. When the input to the FET `gate`, which comes from the $\overline{Q}$ output of the D-type flip-flop, is `LOW`, then the FET is switched off, and it draws no current. However, when $\overline{Q}$ goes `HIGH`, the FET pulls the + input of the comparator to ground, and holds it there. The box marked "osc" represents a typical digital clock. The arrow within the circle connected to $+V_{cc}$ is the symbol for a "current source", which means that its output is a constant current, regardless of the impedance at its output (within reasonable bounds).

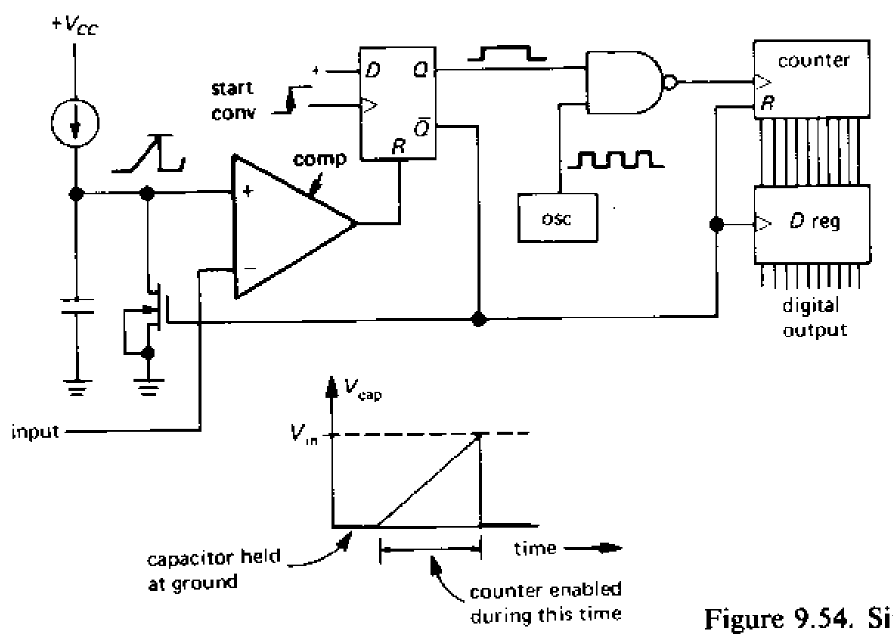

Figure 32: Scheme for single-slope ADC, from text.

The process begins when a rising-edge signal is sent to the flip-flop, for example from a debounced switch. Since the D input is `HIGH`, then Q goes `HIGH`. Hence the counter, no longer being held at reset by the flip-flop, begins counting. At the same time the FET is switched off and a signal is sent to the $-$ input of the comparator. Now we must analyze the nature of this signal.

The voltage across a capacitor V_{cap}, is related to its stored charge by $V_{\text{cap}} = Q/C$, where C is the capacitance. Differentiating gives $dV_C/dt = I/C$. Now, because of the current source, the right-hand side of this equation is a constant. Finally, since one side of the capacitor is at ground, then the comparator $+$ input is just V_{cap}. Hence, we can integrate our expression over a time interval Δt to give:

$$V_+ = V_{\text{cap}} = (I/C)\Delta t$$

Since I/C is a known constant, this equation allows one to convert the V_+ input to a time Δt to be measured by the counter. This linear relation between V_+ ($= V_{\text{cap}}$) and Δt is illustrated in the figure. The counter stops (is reset) and its final count stored in the register when V_+ becomes equal to V_{in}, thus changing the state of the comparator. This also resets the flip-flop, thus returning the circuit to its initial state.

The dual-slope ADCs work similarly, but with a two-step process. First, a capacitor is charged for a fixed time τ with a current source whose current is proportional to V_{in}, $I = \alpha V_{\text{in}}$, where α is the constant of proportionality. Hence, V_{cap} is proportional to τ: $V_{\text{cap}} = \alpha V_{\text{in}}\tau/C$. The capacitor is then discharged at constant current I' and the time Δt to do so is measured. Therefore,

$$\Delta t = [C/I']\,[\alpha\tau/C]\,V_{\text{in}} = \beta V_{\text{in}}$$

where $\beta = \alpha\tau/I'$ is a known constant.

This technique has two advantages compared with single-slope. First, we see from the equation above that the result is independent of C. This is good, as precise capacitance values are difficult to fabricate. Second, the integration of the input voltage in the charge-up step allows 60 Hz pickup noise (or other periodic noise) to be averaged to zero.