Converters:

a) Digital/Analog Converters:

(1)

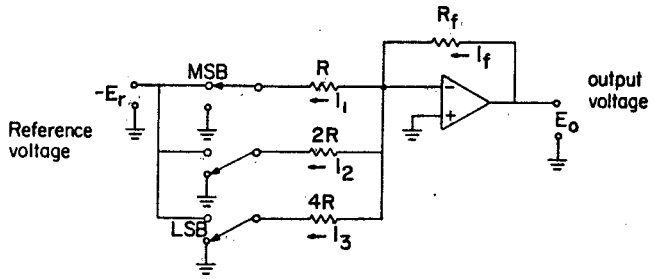


Fig. 2-19. A weighted-resistor three-bit D/A converter.

(2)

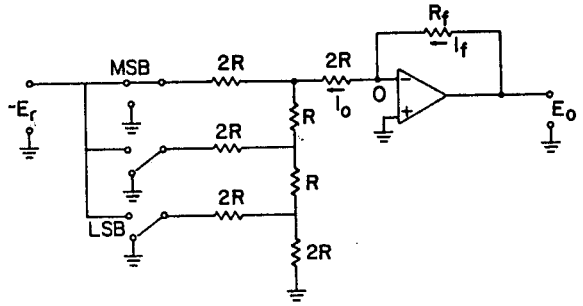


Fig. 2-20. An R-2R ladder D/A.

The second type is more common.
⇒ Resistors are only of two types, and can therefore better be matched.
⇒ Cheaper to get higher accuracy.

D/A converters are <u>cheap and fast</u>.
Usually 8 bits or 16 bits.

b) Analog/Digital Converters.

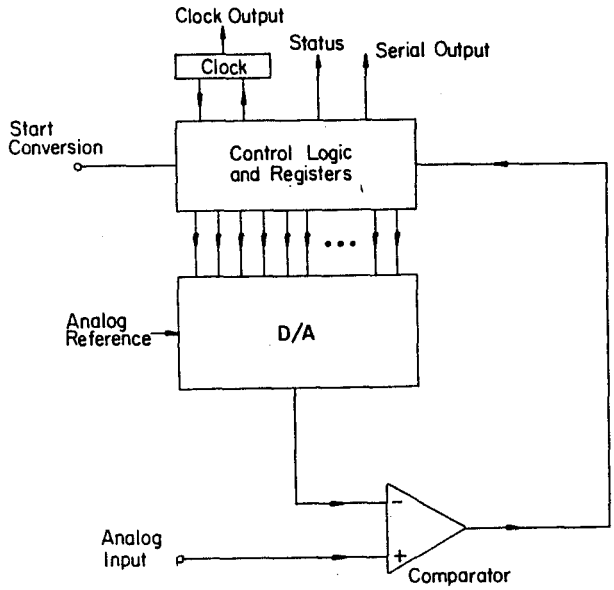
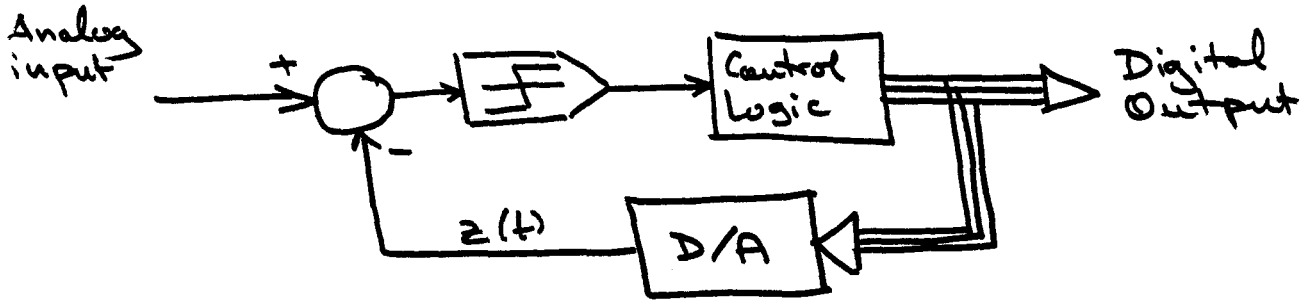


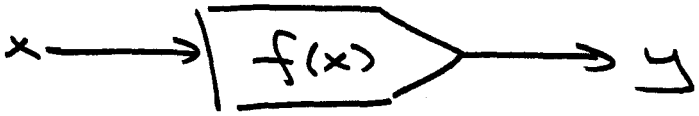
Fig. 2-25. Simplified block diagram of a typical successive-approximation A/D.

An A/D converter is by itself a control circuit:

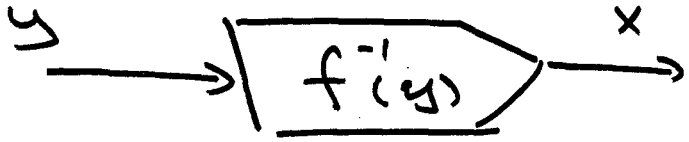


This is an "old trick" used often during the times of analog computers.

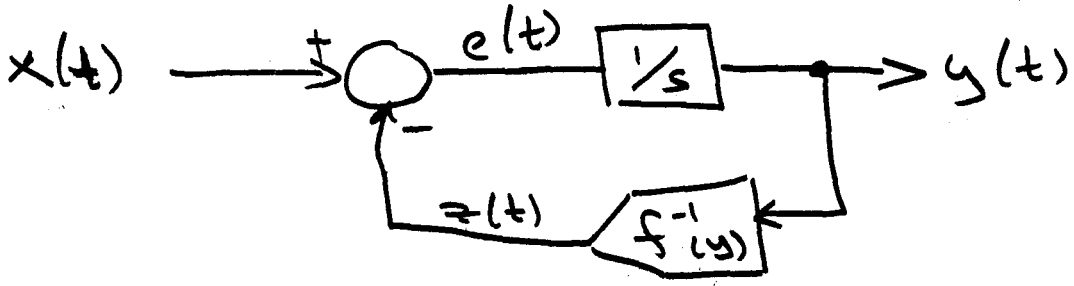
If a particular transfer characteristic



was to be built, but only the inverse function



was physically available, one used to build a control circuit of the type:



As long as $e(t) \neq 0$, the integrator keeps working. Thus, in the steady-state,

$e(t) \equiv 0 \implies z(t) \equiv x(t) \implies y(t) = f(z(t))$

as desired. (If the circuit is stable.)

The A/D-circuit basically works along the same lines. The "digital logic" block might be a BCD counter that counts upwards as long as the feedback signal $z(t)$ is too small, and downward otherwise, or a special-purpose circuit that works as follows:

(a) Set the most significant bit (MSB) to one, all others to zero. This gives 1/2 of full scale (FS) output on $z(t)$.

(b) If $z(t) >$ input, set MSB back to zero, and try next bit $\Rightarrow$ 1/4 of FS, but

If $z(t) <$ input, leave MSB at one, and try next bit $\Rightarrow$ 3/4 of FS.

(c) Continue until the least significant bit (LSB) is reached.

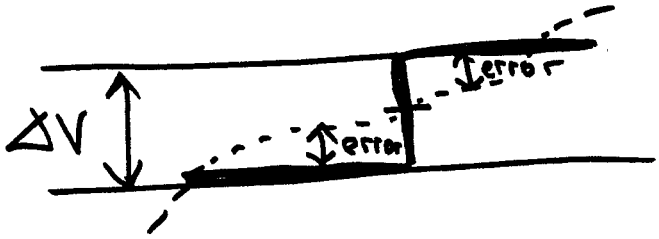
This is the most conventional algorithm used in commercial A/D's.

Quantization error:

With a n-bit converter, 2^n different "numbers" can be represented. Thus, if they are chosen to be equidistantly distributed over the FS,

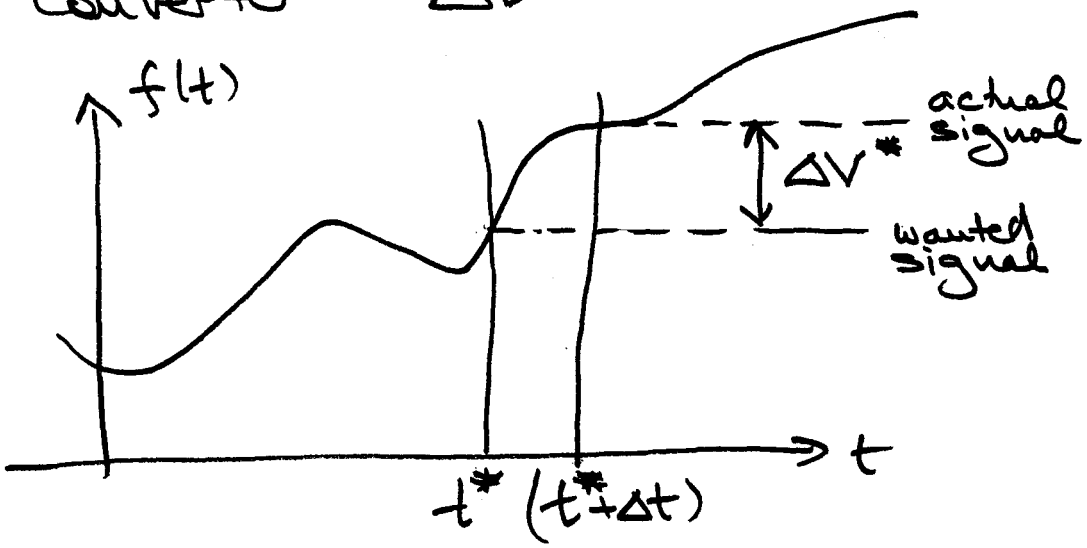
$$\Rightarrow \Delta V = 2q = \frac{V_{FS}}{2^n}$$

The "error" will thus always be smaller than $\Delta V/2$



We call this quantity q (quantization error).

Obviously, the conversion takes time. Let us call the delay of the A/D-converter Δt



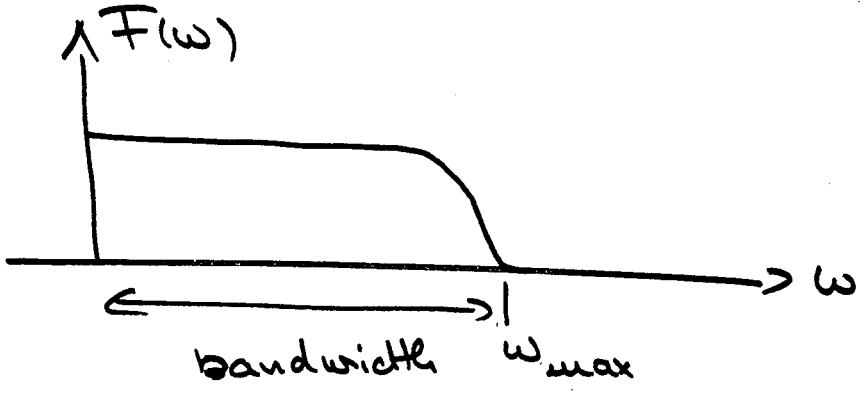
$$\Rightarrow \quad \Delta V^* \approx \left.\frac{\partial f(t)}{\partial t}\right|_{t=t^*} \cdot \Delta t$$

We want to guarantee that $\underline{\Delta V^* \leq q}$

$$\Rightarrow \Delta V^* = \left.\frac{\partial f(t)}{\partial t}\right|_{t=t^*} \cdot \Delta t \leq q = \frac{V_{FS}}{2^{n+1}}$$

$$\Rightarrow \quad \left.\frac{\partial f(t)}{\partial t}\right|_{max} = \left| \frac{V_{FS}}{2^{n+1} \cdot \Delta t} \right|$$

Let us consider the Fourier spectrum of $f(t)$



$$\Rightarrow f_{\omega_{max}}(t) = \frac{V_{FS}}{2} \cdot \sin(\omega_{max} t)$$

$$\Rightarrow \frac{\partial f_{\omega_{max}}}{\partial t} = \frac{V_{FS}}{2} \cdot \omega_{max} \cdot \cos(\omega_{max} \cdot t)$$

$$\Rightarrow \left.\frac{\partial f}{\partial t}\right|_{max} = \frac{V_{FS}}{2} \cdot \omega_{max}$$

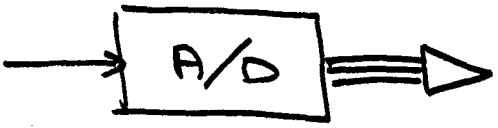
$$\Rightarrow \underline{\omega_{max} \leq \frac{1}{\Delta t \cdot 2^n}}$$

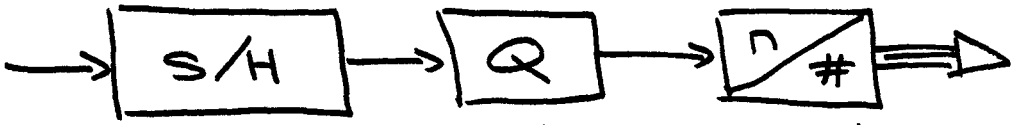
⇒ The better the precision of the converter, the slower the signal must be to be converted correctly.

It is usually preferred to receive the correct signal with a fixed delay, rather than an incorrect signal sooner.

⇒ Always put a sample and hold circuit in the input of an A/D converter, with a sampling time > conversion time. Then the problem disappears at the price of a delayed signal. (Easier to analyze.)

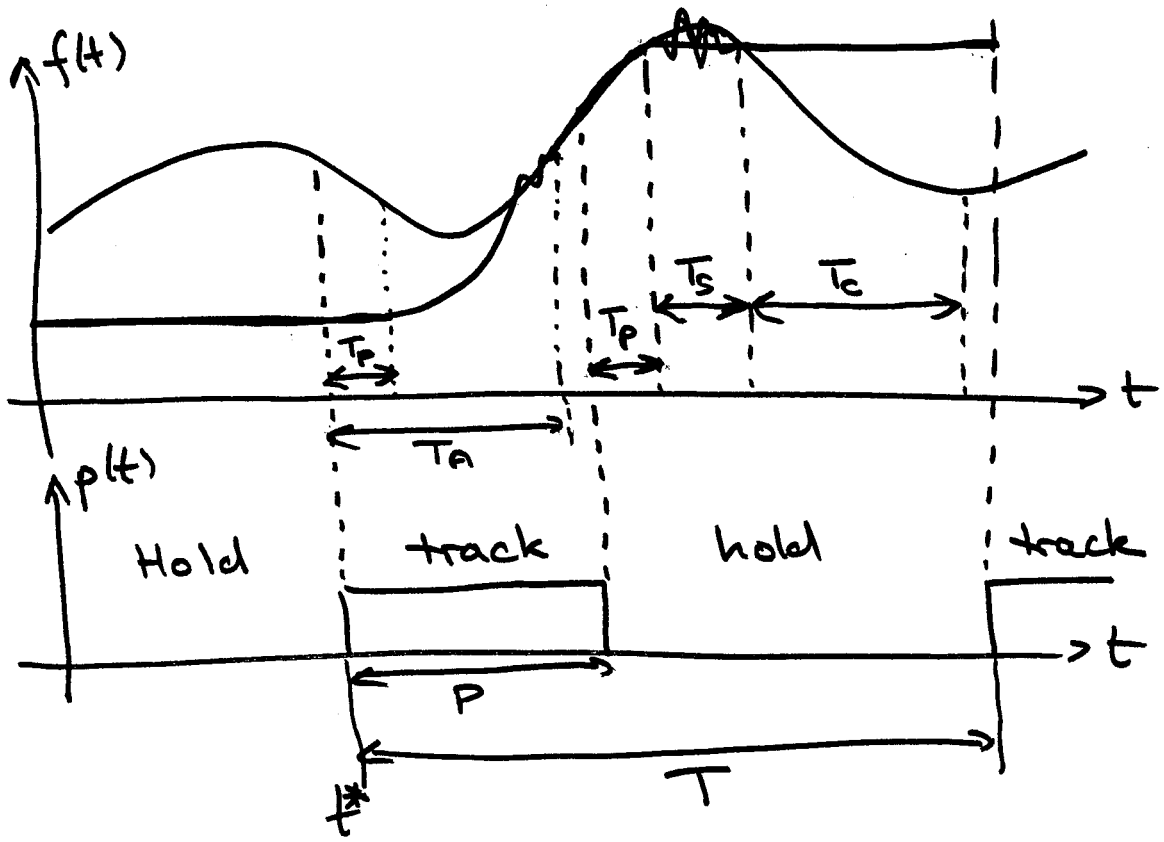
→





Sample & Hold circuit:

Let us analyze now the effects of real rather than idealized S/H's:



p ::= pulse time

T ::= sampling time

T_P ::= Aperture time

T_A ::= Acquisition time

T_S ::= Settling time

T_C ::= Conversion time

Conditions:

$$p > T_A$$

$$T > p + T_p + T_s + T_c$$

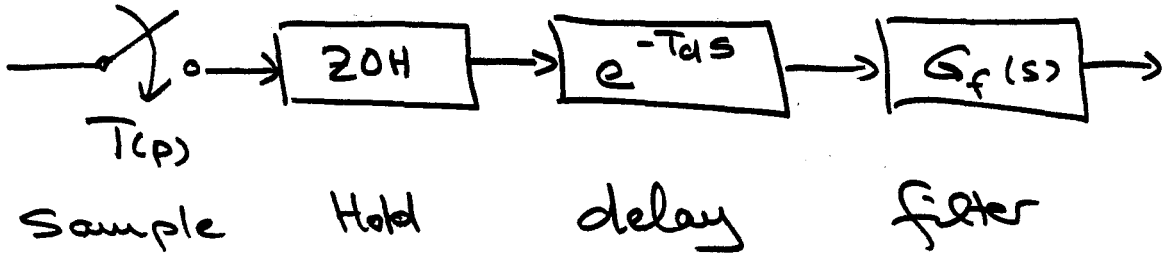
A second condition for T comes from the computing time for the discrete controller. Obviously

$$T > T_{comp}$$

At the output of the A/D converter, we shall receive the signal at time $t^* + \underbrace{p + T_p}_{\Delta T_1}$

delayed by: $t^* + \underbrace{p + T_p + T_s + T_c}_{\Delta T_2}$

At the output of the D/A converter, we shall see the signal as shown above which can be idealized as:



On some data sheets for converters, the delay time (T_d) and the dynamic characteristics ($G_f(s)$) are specified.

Usually, we are going to ignore those effects, as the converters are supposed to be much "faster" than the control circuits for which they are used, and since the control circuits usually have low pass characteristics.

Digital Signals:

a) Positive Integers:

⇒ Binary representation
word length = n
⇒ 2^n numbers can be represented

b) Integers:

⇒ first bit taken as sign bit

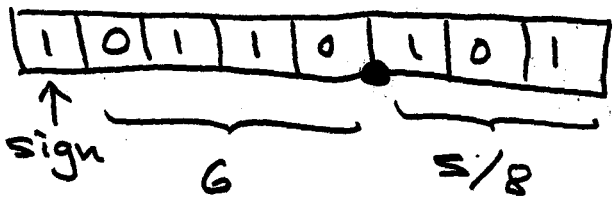
→ numbers $\in \pm 2^{n-1}$

This is one number less than before, because $(+0) \neq (-0)$.

This problem can easily be solved by using another representation for negative numbers.

c) Fixed-point Real:

e.g.



$$\Rightarrow \text{number} = -6.625$$

There are obviously still 2^n "different" numbers possible, but the resolution is now:

$$\Delta V = 1/8$$

and the total range is

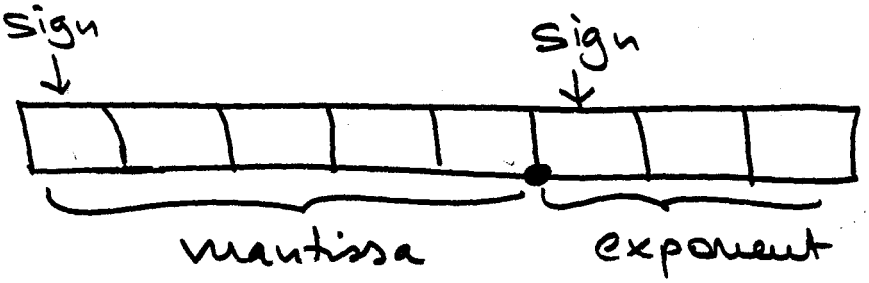
$$V_{FS} = \pm 16$$

$$\Rightarrow \Delta V = \frac{V_{FS}}{2^n} = \frac{32}{2^8} = \frac{32}{256} = \frac{1}{8} \quad \checkmark$$

Fixed-point real numbers are really nothing new. Just a reinterpretation of the previous data representations.

d) Floating point Real:

e.g.



"smallest" number: $1 \cdot 10^{-3} = 1/1000$

"largest" number: $15 \cdot 10^{+3} = 15'000$

There suddenly seem to be more numbers. But this is not correct.

Numbers would be:

$1 \cdot 10^{-3}$
$2 \cdot 10^{-3}$
⋮
$15 \cdot 10^{-3}$
$1 \cdot 10^{-2}$
$2 \cdot 10^{-2}$
⋮
$15 \cdot 10^{-2}$
⋮

⇒ There are still 256 numbers, but no longer equidistantly spaced.

This is bad for error analysis, and is only done where the accuracy is sufficiently high that it does not matter ⇒ at least 32 bits.

⇒ Never used inside a converter. Only inside the computer, because the scaling problem gets easier
⇒ reduction of software cost at the price of lower speed per Dollar.