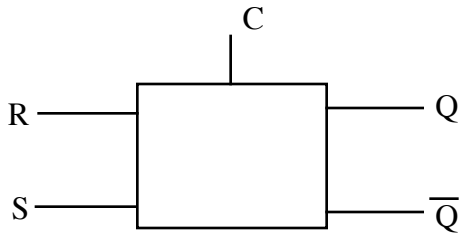
Review from last week:
Flip-Flops:
Basic counting unit in computer
counters
shift registers
memory

Example: RS flip-flop or Reset-Set flip-flop

Flip-flops, like logic gates are defined by their truth table. Flip-flops are controlled by an external clock pulse.

R	S	Q_{n+1}
0	0	Q_n
1	0	0
0	1	1
1	1	undefined

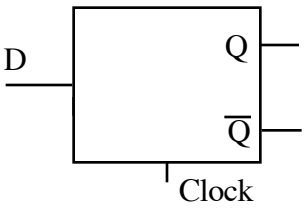


Q_n is the present state of the FF and Q_{n+1} will be the output after the clock enables the FF to look at its inputs (R and S).

Many FF's change state (Q_n Q_{n+1}) on the trailing edge of the clock.

Note: *The state with* R = S =1 *is undefined. The output is not predictable!*

Example: D flip-flop (Like RS but only one input)

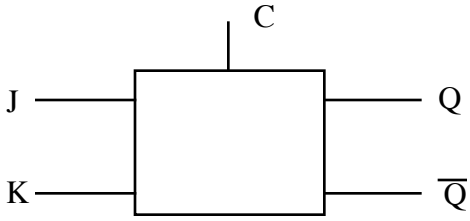


D	Q	Q_{next}
0	0	0
0	1	0
1	0	1
1	1	1

Example: JK flip-flop

JKFF is like the RSFF except that both inputs (J and K) can be high (1).

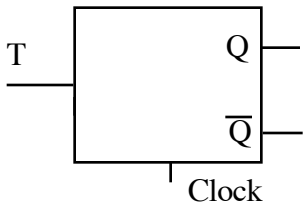
J	K	Q_{n+1}
0	0	Q_n
1	0	1
0	1	0
1	1	$\overline{Q}_n$



Most JKFF's have a connection for forcing Q = 0 (reset) or forcing Q = 1 (set).

Example: T (*Toggle*) flip-flop

T flip-flop is like the JKFF with both inputs (J and K) tied to each other.



T	Q	Q_{next}
0	0	0
0	1	1
1	0	1
1	1	0

- A few words about clocking the flip-flops and timing of inputs.

  Setup time:

  For each type of flip-flop there is a minimum specified time relative to the clock pulse during which time the input(s) to the FF must be stable (i.e. not change logic levels).

  Hold time:

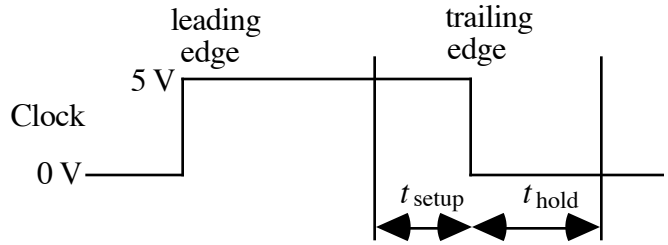
  For each type of flip-flop there is a minimum specified time after Q changes state that the input(s) to the FF must be stable (i.e. not change logic levels).

Example: 74LS112 JK flip-flop (the one we use in lab)

This FF changes state (Q) relative to the trailing edge of the clock.

The setup time is 20 nsec (2×10^{-8} sec) while the hold time is 0 nsec.

The maximum clock speed of this FF is 30 MHz.



Thus the data on J and K must be stable for at least $t_{setup} + t_{hold}$.

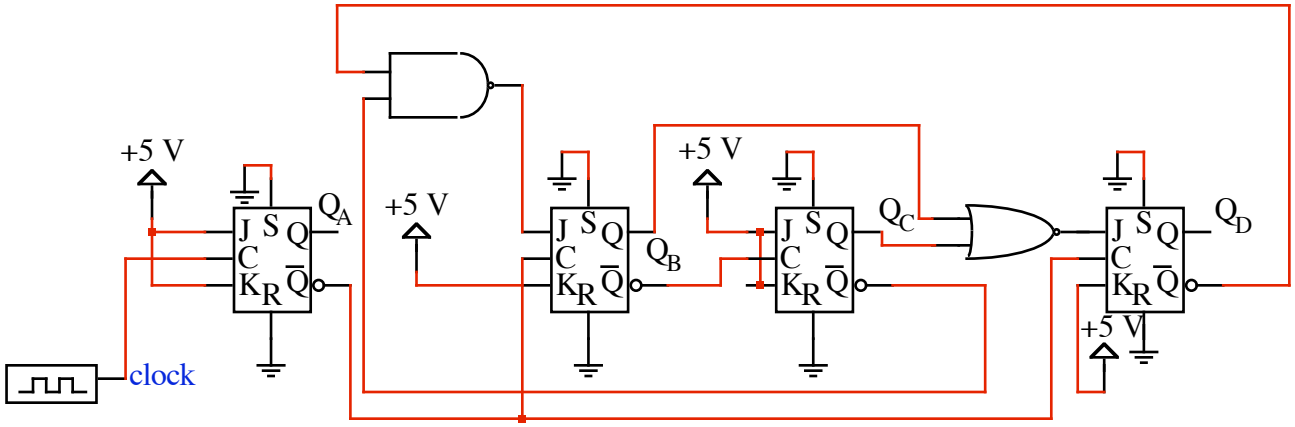
- Sometimes circuits with flip-flops are classified according to how the clock is distributed to the FF's.

There are two clocking schemes:

Synchronous: All FF's are clocked at the same time. The easiest way to do this is to use one clock and distribute it to all the FF's.

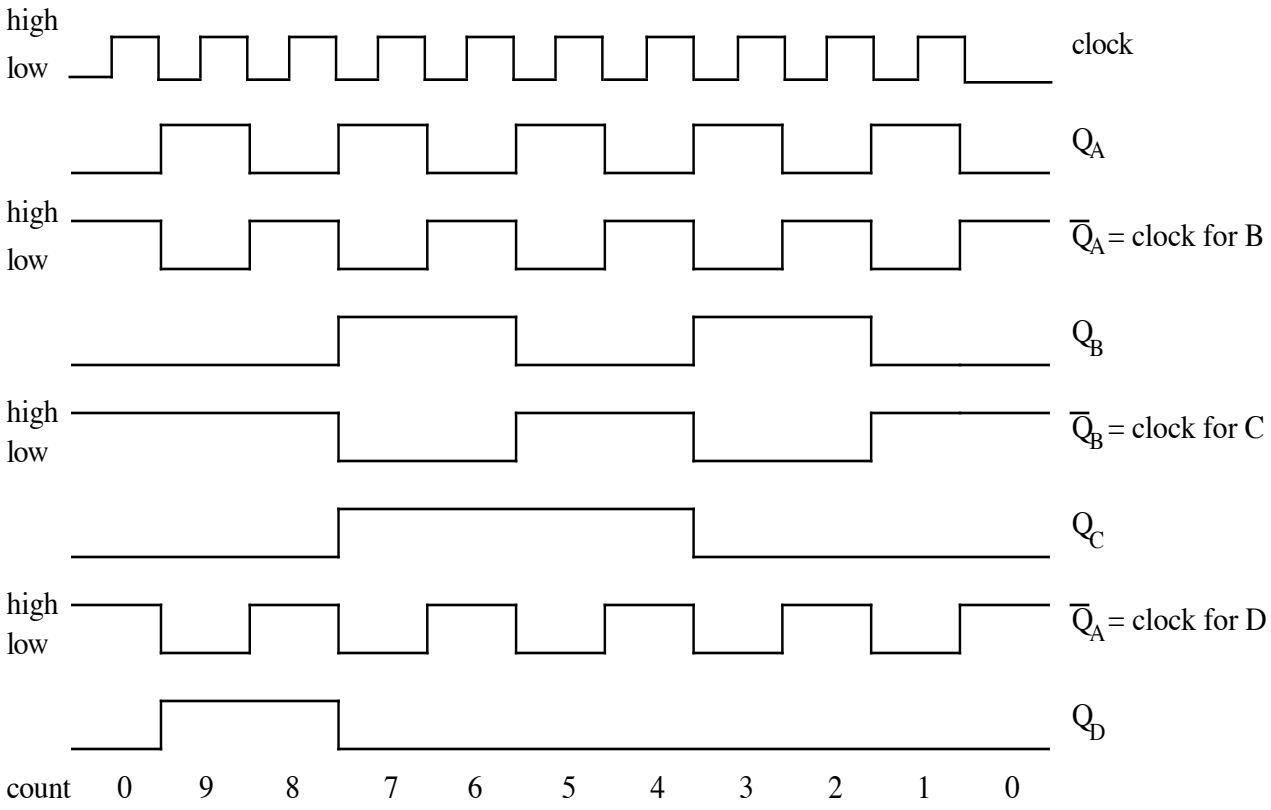
Asynchronous: FF's are clocked at different times, usually by different clocks. Last week's example was an example of this type of circuit. The first FF was clocked by a "clock", while the second FF was clocked by the output (Q) of the first FF.

Example: Divide by 10 ripple down counter (counts from 9 to zero)
Asynchronous counter



$J_A = 1$ $\quad J_B = \overline{\overline{Q}_C \overline{Q}_D} = Q_C + Q_D$ $\quad J_C = 1$ $\quad J_D = \overline{Q_B + Q_C} = \overline{Q}_B \overline{Q}_C$

$K_A = 1$ $\quad K_B = 1$ $\quad K_C = 1$ $\quad K_D = 1$

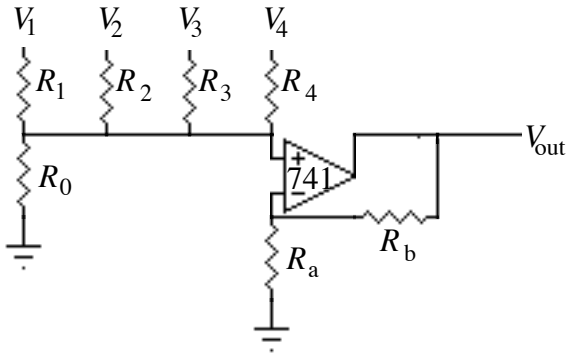


Conversion between Analog and Digital Signals

• Digital to Analog Conversion (DAC): There are two simple circuits commonly used to convert a digital signal to an analog voltage.

Weighted Resistor Ladder: The circuit is shown below.

We assume that the input voltages (V_1, V_2, V_3, and V_4) are logic levels. For this example let us use TTL levels and assume a high = 3 V and a low = 0 V.



The output voltage is given by:

$$V_{out} = \frac{\frac{R_b}{R_a} + 1}{\frac{1}{R_0} + \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} + \frac{1}{R_4}} \left(\frac{V_1}{R_1} + \frac{V_2}{R_2} + \frac{V_3}{R_3} + \frac{V_4}{R_4} \right)$$

If we choose the resistors as follows:

$R_1 = R_a = 1$ kΩ, $R_2 = 2$ kΩ, $R_3 = 4$ kΩ, $R_4 = R_0 = 8$ kΩ and $R_b = 15$ kΩ,

then we get the following simple relationship for V_{out}:

$$V_{out} = 8V_1 + 4V_2 + 2V_3 + V_4$$

Thus if V_{in} represents a binary number (e.g. 1001 = $V_1V_2V_3V_4$ with V_1 being the highest order bit) then the output voltage varies from 0 to 45 Volts (remember $V_1..V_4$ are all either 0 or 3 V). Therefore the digital input 1001 has an analog output of 27 V = (3x8 + 3) V.

Unfortunately there are several bad points with this conversion scheme:

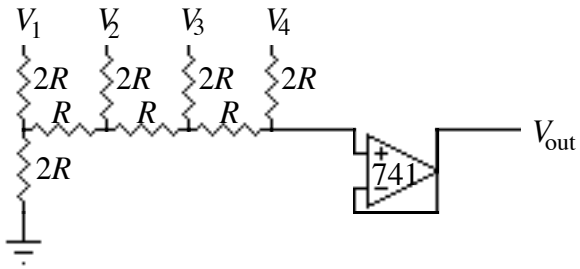
a) the output can be a large voltage (e.g. 45 V)

b) circuit needs 5 high precision resistors (expensive)

c) the current (and therefore power) in the resistors varies by 15.

The following circuit fixes up many of these problems:

Binary Ladder Network (R-2R Network):



The output voltage for this circuit is:

$$V_{out} = V_1/2 + V_2/4 + V_3/8 + V_4/16$$

This circuit needs only 2 values of precision resistors compared with the 5 of the previous design. Also the power dissipated in the resistors only varies by a factor of 2, compared with the previous factor of 15.

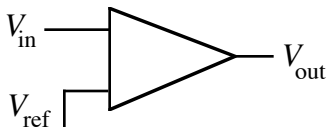
There are still some bad points:
a) still need precision components
b) the output voltage will usually be a fraction of the input (low noise immunity)

For example if $V_{in} = 1001$, then $V_{out} = 3/2 + 0/4 + 0/8 + 3/16 = 27/16$ V for TTL logic levels.

• Analog to Digital Conversion (ADC)

Parallel A to D conversion ("Flash Encoder" or Flash ADC)
very fast and very simple method
use comparators for the conversion

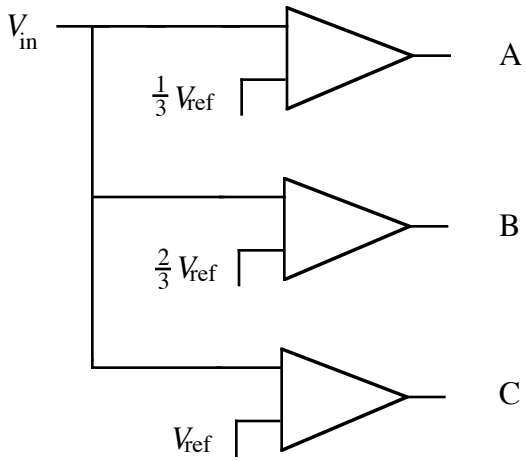
Example: one bit ADC using one comparator



$V_{out} = 1$ (high) if $V_{in} > V_{ref}$
$V_{out} = 0$ (low) if $V_{in} < V_{ref}$

How many comparators do we need for a given accuracy?
Suppose we want to convert an analog number into a 2 bit digital number.
For 2 bits there are 4 possible outcomes (00, 01, 10, 11), it takes 3 comparators.

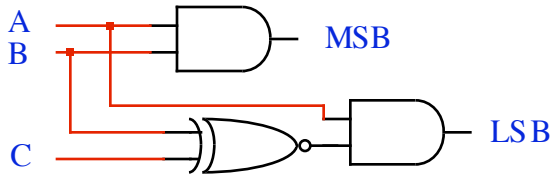
Example: 2 bit parallel converter



Truth Table

V_{in}	A	B	C	Output
$< \frac{1}{3}V_R$	0	0	0	0
$\frac{1}{3}V_R < V_{in} < \frac{2}{3}V_R$	1	0	0	1
$\frac{2}{3}V_R < V_{in} < V_R$	1	1	0	2
$> V_R$	1	1	1	3

Logic gates are needed to implement the truth table:



$$LSB = A\overline{B}\,\overline{C} + ABC = A(\overline{B}\,\overline{C} + BC)$$

$$MSB = AB\overline{C} + ABC = AB(C + \overline{C}) = AB$$

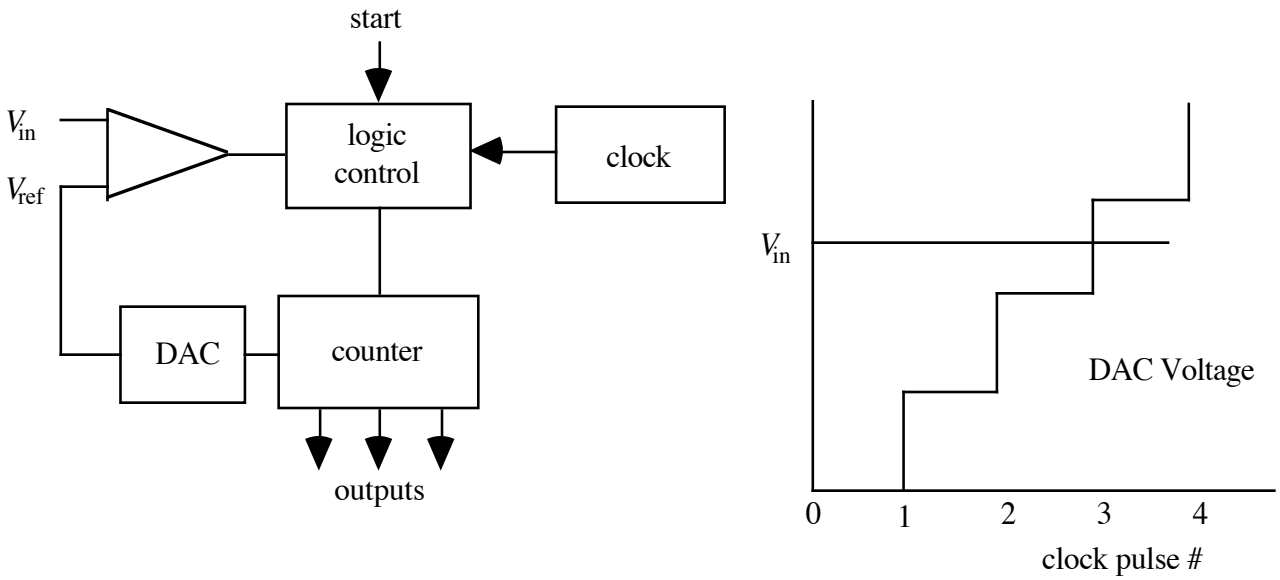
There are several problems with this scheme:

1) need lots of comparators: 2^n - 1 for n bit accuracy.
   Suppose we want 1 part in 1000 accuracy (0.1%), it takes 10 bits
   $= 2^{10} - 1 = 1023$ comparators!
2) the number of logic gates necessary to code the output is large and the logic gets very complicated.

Counter ADC (staircase method):

Good news: only uses one comparator

Bad news: much more complicated than parallel method



When $V_{DAC} > V_{in}$ the logic circuit stops the clock and the counter outputs a binary number which is just the number of clock pulses.

The DAC could be:

an integrator

a resistor ladder

a voltage reference

Problems with this system:

a) control logic is complicated (use microprocessors + gates +...)

b) time to digitize depends on V_{in}.

Example: suppose clock runs at 5 MHz, and you want 10 bit accuracy.

10 bits = 1024 clock pulses.

Can only digitize at about 5 kHz, *which is fairly slow!*

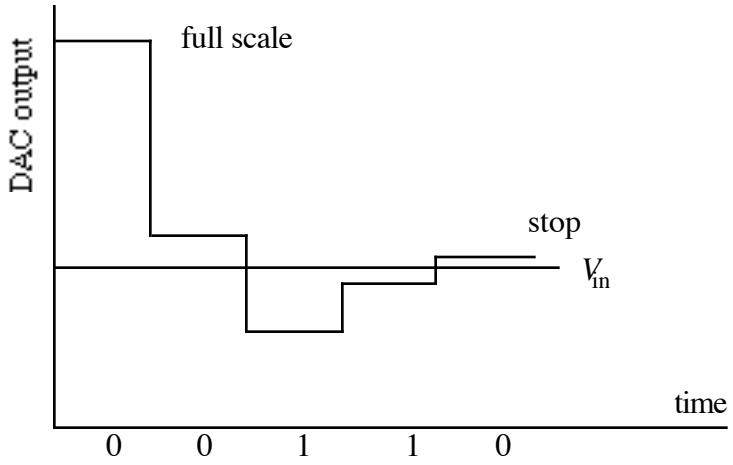
Successive Approximation ADC:

control logic is very complicated, but easy to program, like a binary search.
conversion time is almost independent of V_{in}, very fast.

Example: 5 bit ADC:

The circuit:

outputs 0 when $V_{DAC} > V_{in}$ (steps 1, 2, 5)
outputs 1 when $V_{DAC} < V_{in}$ (steps 2, 3)



In order to convert V_{in} to a binary number the following steps are followed:

a) turn on MSB in DAC and compare with V_{in}.
0 if V_{in} < MSB
1 if V_{in} > MSB (leave bit on if true)
b) turn on next highest bit, leave bit on if $V_{DAC} > V_{in}$.
c) repeat until least significant bit is checked (5 in this example).

Thus the method requires n comparisons for n bit accuracy.

Note: the staircase approach requires 2^n - 1 comparisons
parallel ADC requires 1 step, independent of accuracy

Time for a 10 bit conversion for the three methods we have considered:
parallel : success Approx. : staircase 1:10:1023