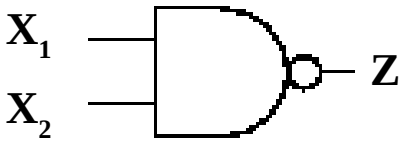
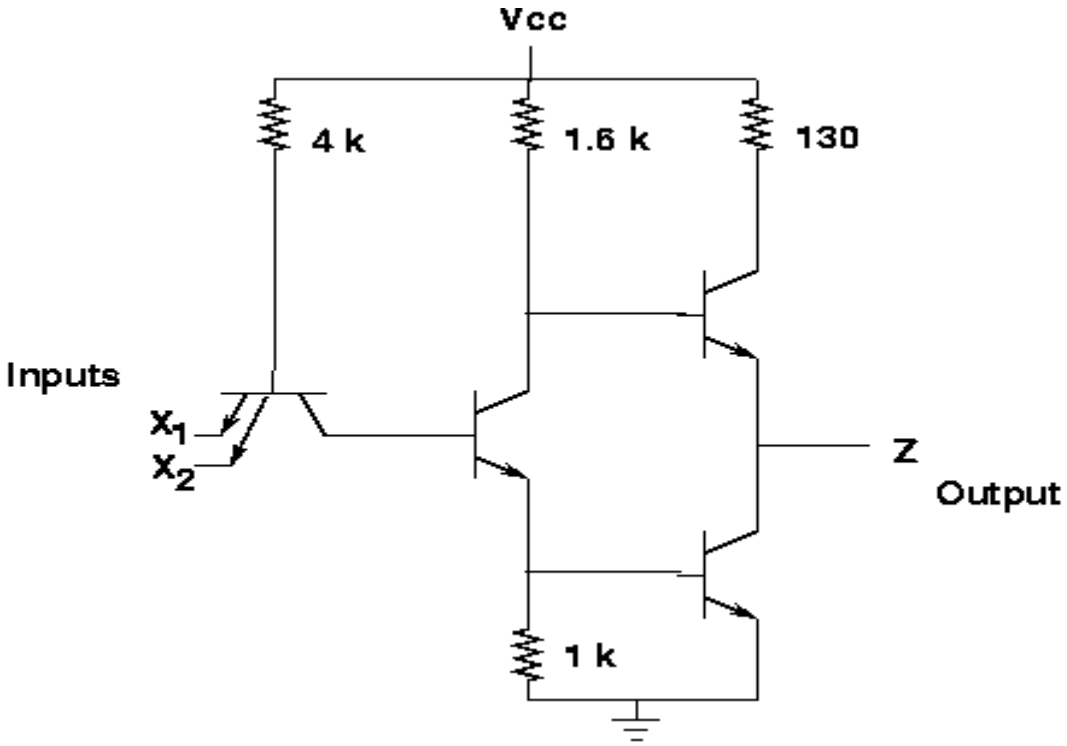
6.111 Lecture # 3



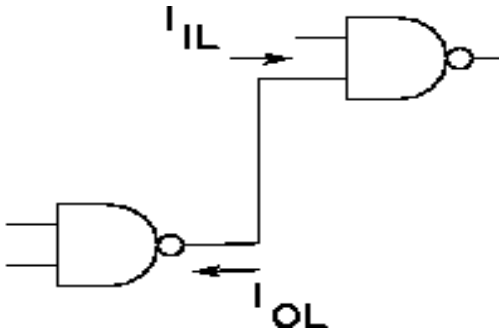


Simplified Schematic of 74LS00 (there are 4 of these in an '00)

Note multiple emitter inputs

Totem pole output

NAND gate is the basic building block



Current Convention

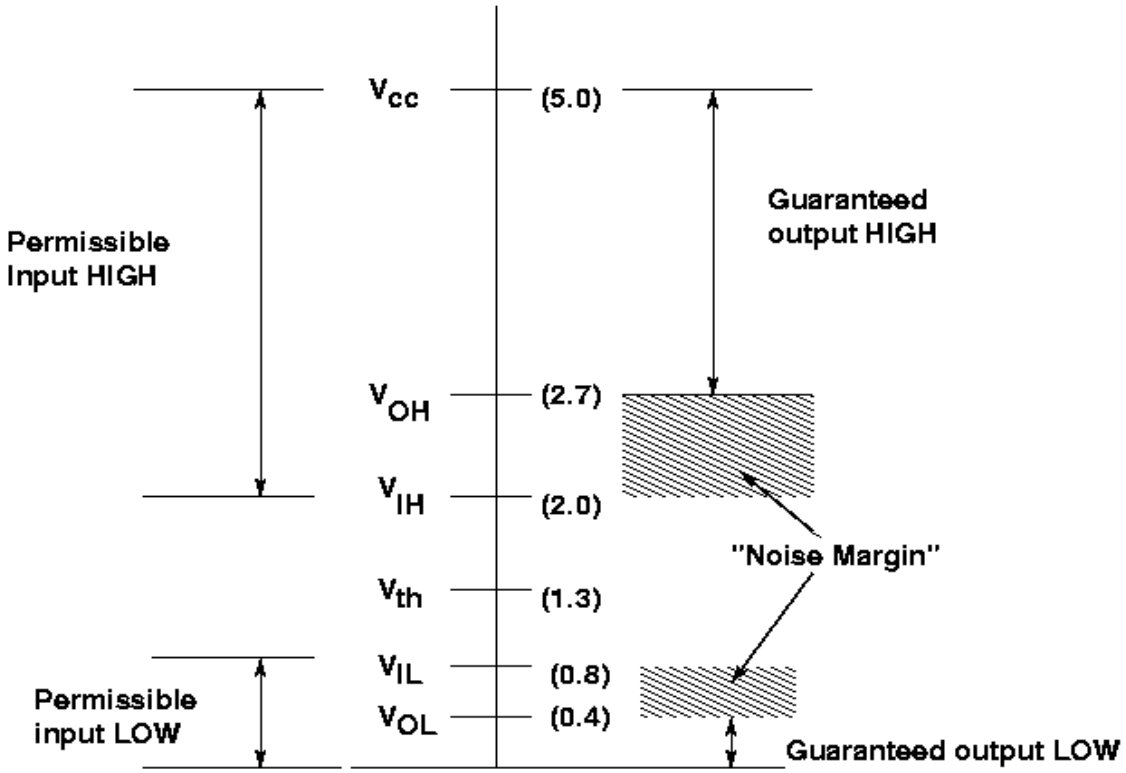
74LS00 **Current (mA)**

Output capability LOW	**I_{OL}**	**8 mA**
Output Capability HIGH	**I_{OH}**	**-400 μA**
Input Required LOW	**I_{IL}**	**-0.4 mA**
Input Required HIGH IIH		**20 μA**

These are typical numbers but read data sheet if in doubt:
There are many exceptions

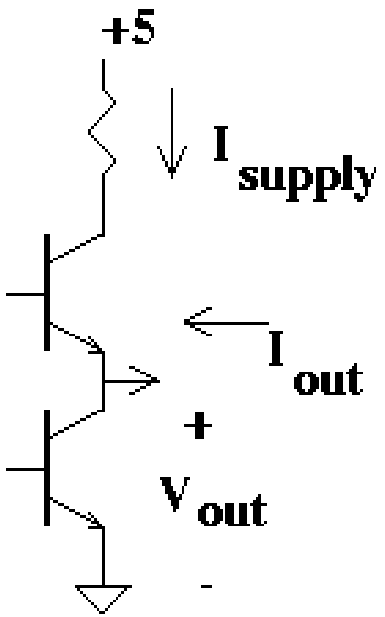
TTL Voltage Ranges

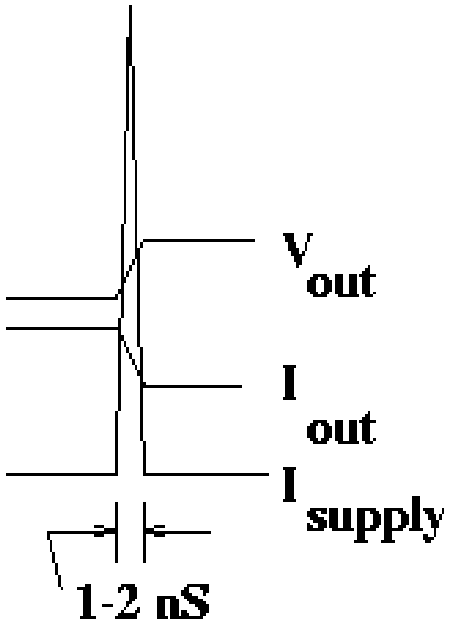
These are important! Valid input and output values are in the ranges shown.



ote that the RUE switching reshold will be fferent for fferent parts or struments -- in ubt, best us a cope

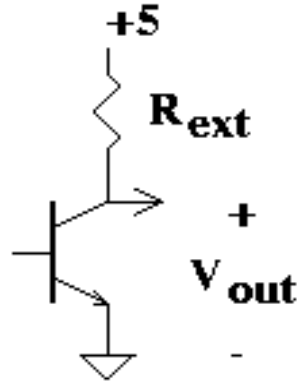
Totem Pole Output (Common for TTL)





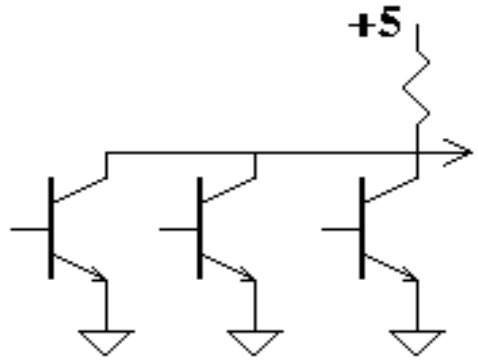
TTL Totem Pole Outputs can draw LARGE current spikes on switching

Some outputs are
open collector: need
a pull-up resistor.
Speed is affected by
R_{ext} and by external
and junction capacitance



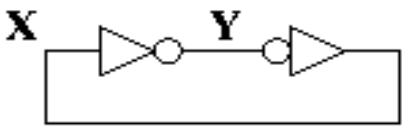
Open collector gates can be
wired together like this to make
'wired AND's.

This is a 'bus' that can be driven
by more than one input source



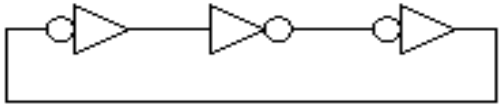
You can't do this with Totem Pole outputs!

Feedback produces 'State'



X	Y
0	1
1	0

Note that either 'state' (X=0, Y=1 or X=1, Y=0) is valid



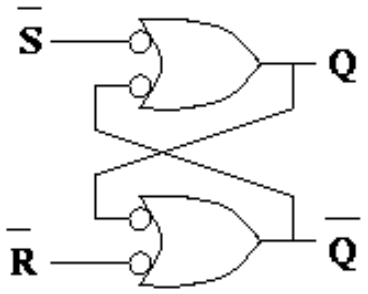
?

Try this in the lab...

What does this one do?

'State' implies memory -- here is how we save information

S-R Latch (74LS279)



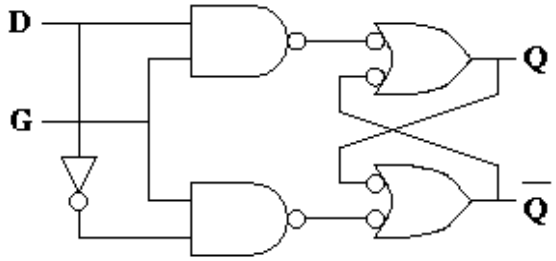
$\overline{S}$	$\overline{R}$	Q	$\overline{Q}$	
0	0	1	1	
0	1	1	0	
1	0	0	1	
1	1	1	0	Both work!
		0	1	(Holds state)

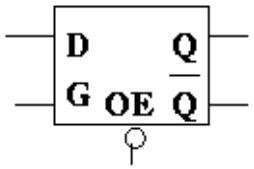
Question: What happens if $\overline{S}$ and $\overline{R}$ go 0 to 1 at the same time?

You can build one of these from NAND gates, but there is a packaged, MSI version.

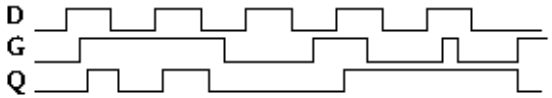
Question: what happens if you build one from NOR gates?

D Latch **(74LS373 is an octal latch with tristate)**





The latch is a "follow and hold:

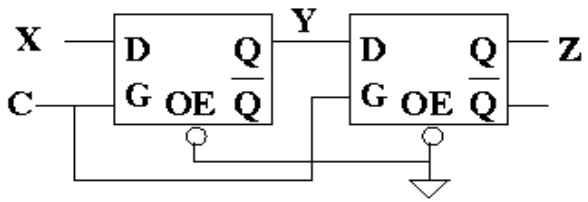


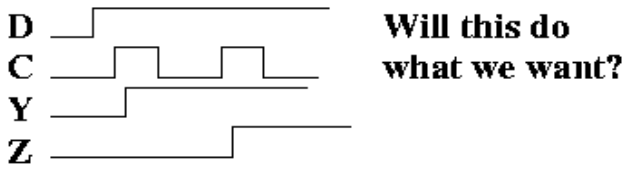
'Latch' is an important notion: its input is controlled by a 'gate'

When the 'gate' goes from high to low, the state of the device holds

Question: what happens if the input and gate change state at nearly the same time?

Problem with latches in multi-stage logic

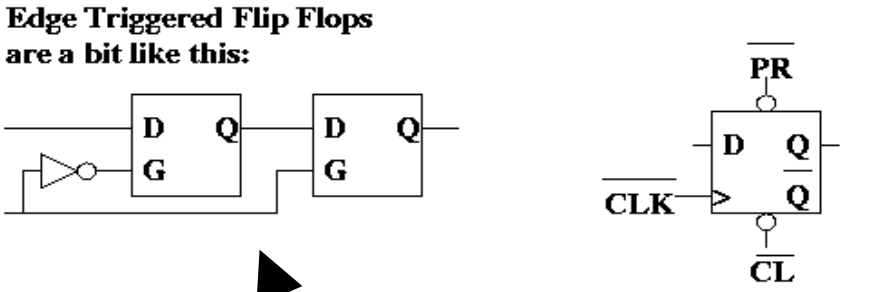


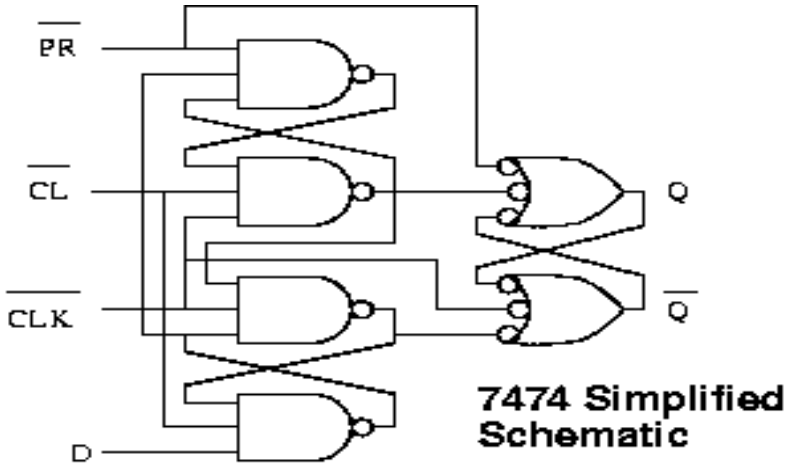


Latch type logic has an issue with propagation of signals

- **How many stages of logic will be affected by a signal change during one clock (G high) cycle?**

Multi-phase clocks have been used for this (Half the G's high one instance, the other half the next), but there is a better solution...

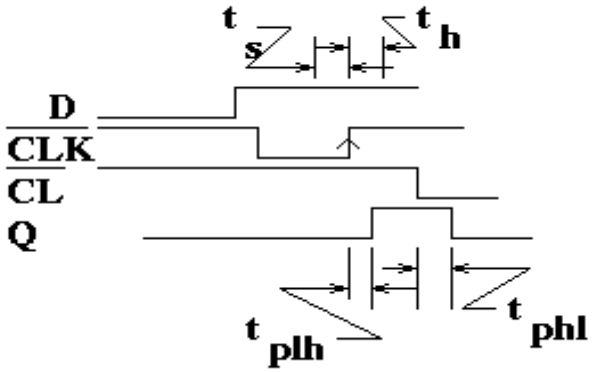




Edge triggered logic differs from latches in that it is the transition of the 'clock' input that causes the flip flop to hold state

Actual implementation is not <u>quite</u> like what is shown here.

It takes a little effort to reason through what this part does. See that the 'preset' and 'clear' are asynchronous, which means they take effect right away, without waiting for the clock edge.

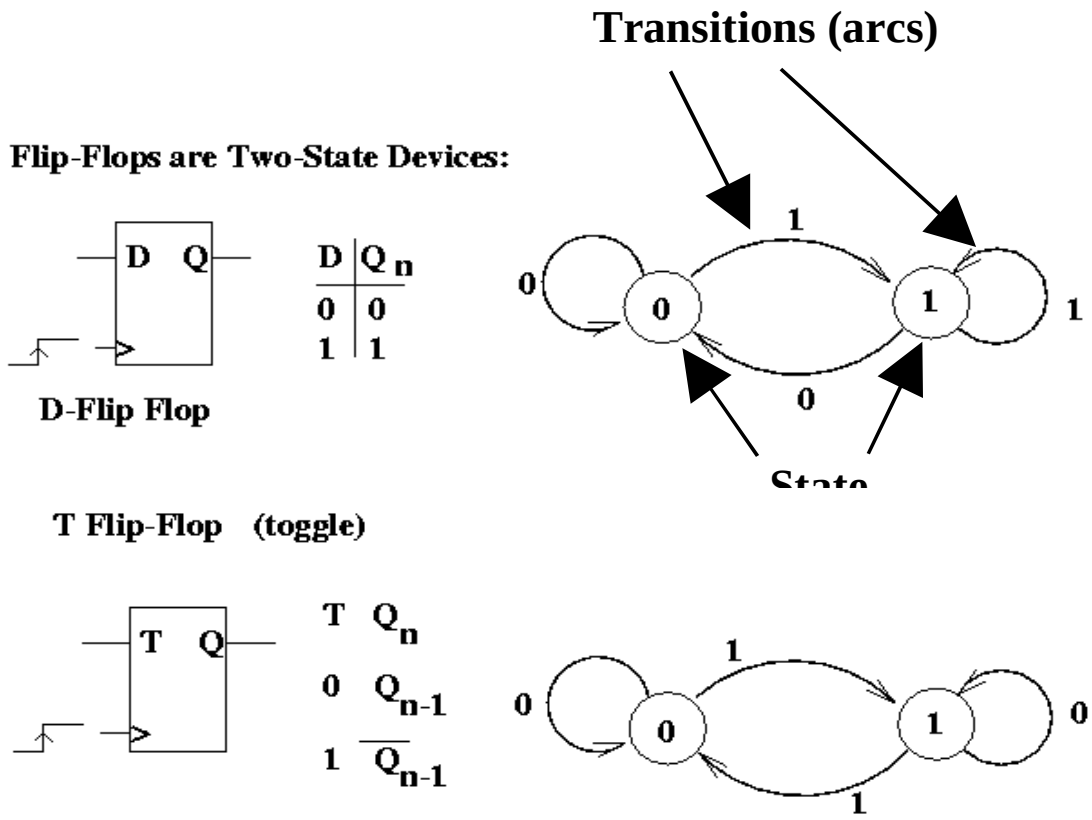


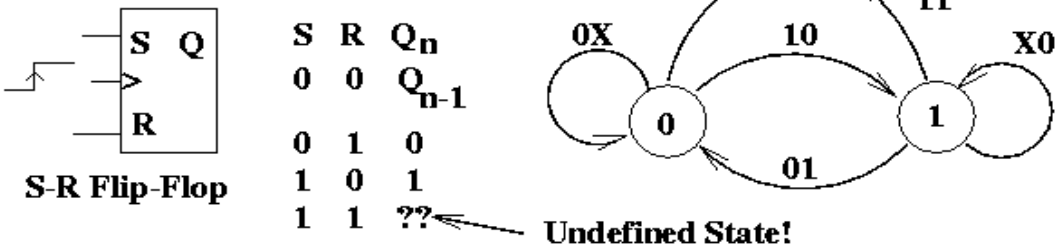
Typical Timing Parameters for 74LS Parts

Setup	t	≥ 20 nS
Hold	t	≥ 5 nS
Clock to Q		≤ 20 nS
CL or PR to Q		≤ 25 nS
CLK high		≥ 25 nS
Max Frequency		25 MHz

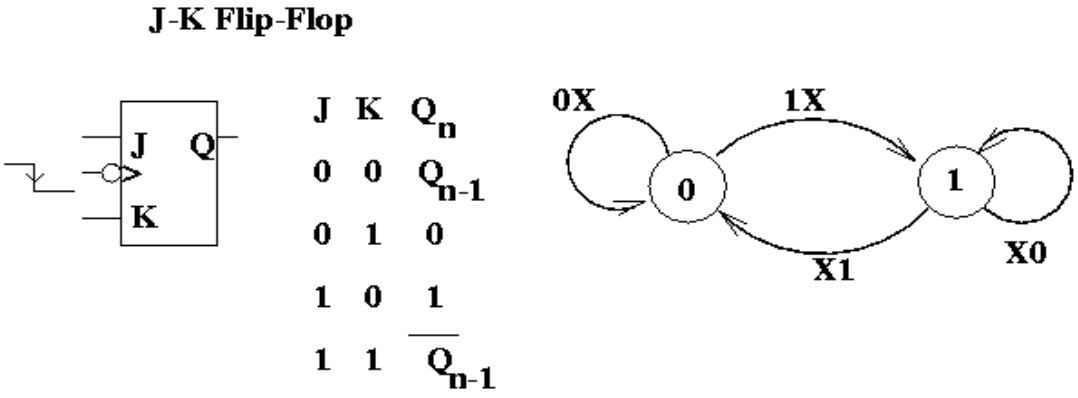
Setup Time: Input must be stable before the clock edge
Hold Time: Input must stay stable after the clock edge
Clock to Q: maximum time for output to be stable after clock edge
CL or PR to Q: maximum time for output to be stable after asynchronous input
Max Frequency = 1/(Clock HIGH + Clock LOW)

Flip flops are simple finite state machines. Here is how we describe such machines

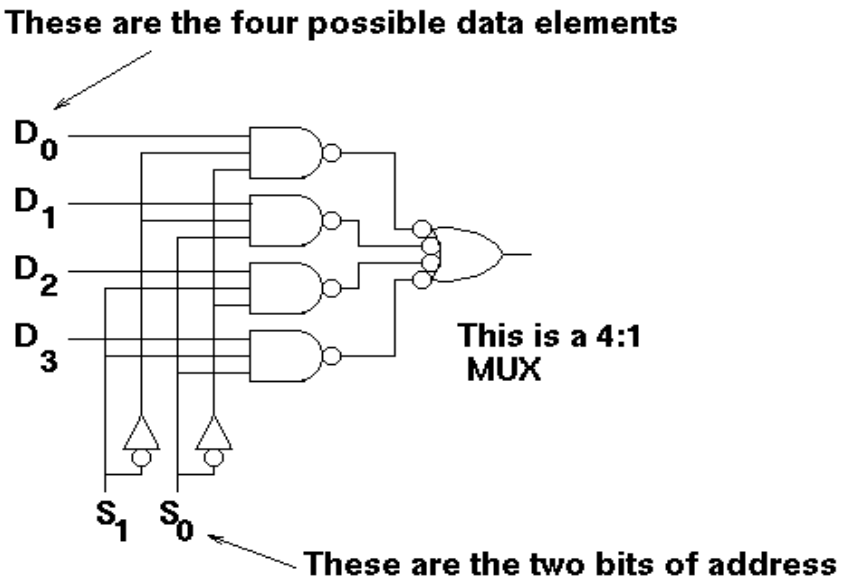




The SR FF is an edge triggered version of the SR latch. It has an undefined state problem that is solved in the JK FF



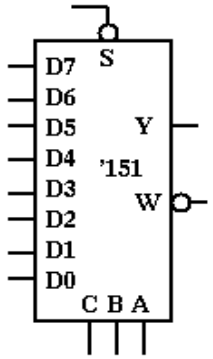
Note this JK has a <u>negative</u> edge triggered clock!

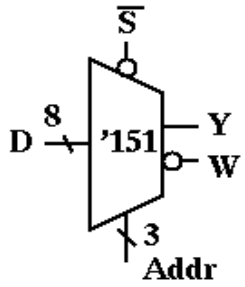


Multiplexer's (MUX'es) are an important building block

This one selects one of four inputs based on an 'address'

74LS151 is a
8:1 MUX
Strobe is
active low
$Y = S * D(CBA)$
$W = \overline{Y}$



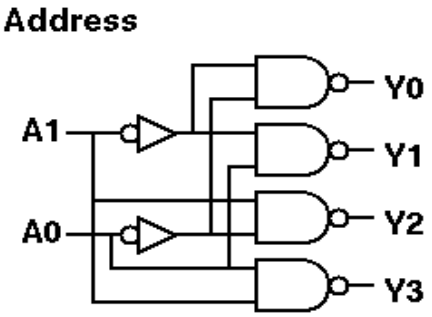


The 74LS151 part has 8 inputs and so 3 bits of address

It also has a 'strobe' input which is functionally a chip select

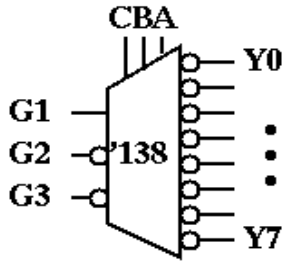
The output is presented both direct and inverted

Demultiplexer or Selector is the inverse of the Multiplexer It selects the addressed line

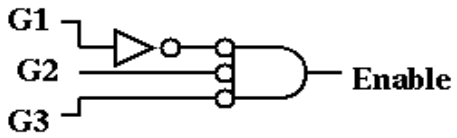


One of these lines is selected (pulled low in this case)

74LS138
3:8 Decoder

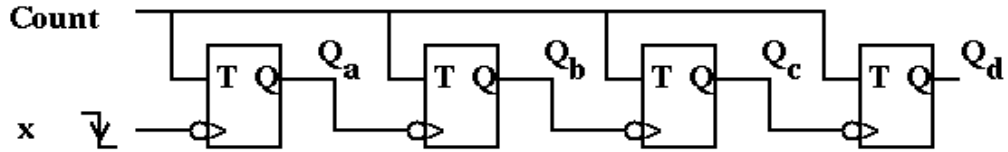


The '138 has a complex enable mechanism



Counting is a very important function in the digital world, and it is done in a variety of ways

Here is a 'ripple' counter using negative edge triggered T flip flops



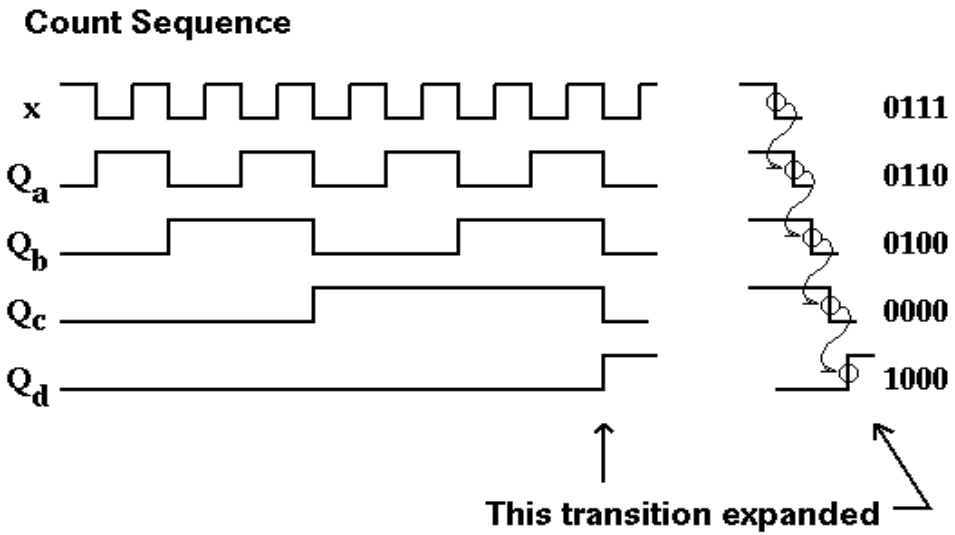
The LSB is on the left in this diagram. It always toggles.

The transition of 1 -> 0 of each 'bit' triggers a toggle of the <u>next</u> most significant bit

Count Sequence:

0 0 0 0
0 0 0 1
0 0 1 0
0 0 1 1
0 1 0 0
0 1 0 1
0 1 1 0
0 1 1 1
1 0 0 0
1 0 0 1
1 0 1 0
1 0 1 1
1 1 0 0

Here is why it is called a 'ripple' counter:



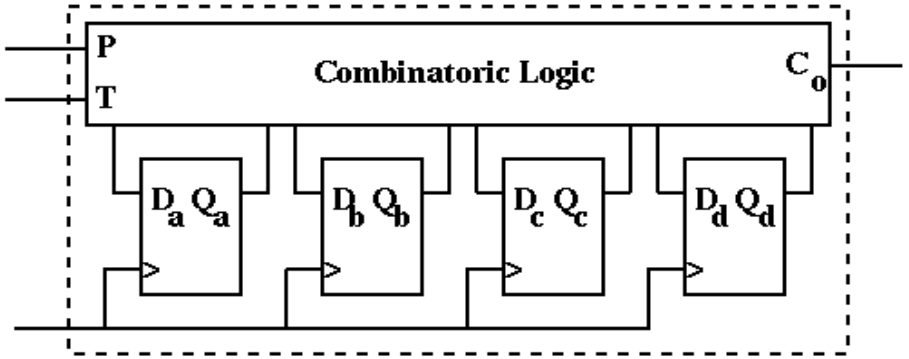
The effect of each input transition must affect all bits, and it does this by rippling through from LSB to MSB

An odd effect is that the transient count is always less than the true count.

Can COUNT fast, but maybe can't be READ fast!

'Synchronous' counters use more logic to reduce the time to stable outputs.

Here is a simplified version of the 4 bit 74LS163 counter



Synchronous Counters: reduce ripple by setting all bits at once

I = P * T
Da = /I * Qa + I * /Qa
Db = /I * Qb + I * Qa * /Qb + /Qa * Qb
Dc = /I * Qc + I * Qa * Qb * /Qc + /Qa * Qc + /Qb * Qc
Dd = /I * Qd + I * Qa * Qb * Qc * /Qd + Qd*/Qa + Qd*/Qb + Qd*/Qc
Rco = T*Qa*Qb*Qc*Qd

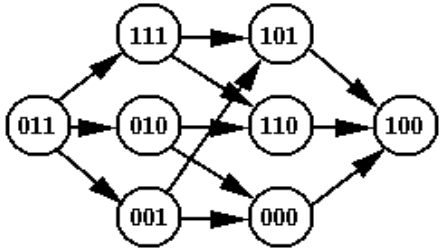
Note that, while all bits of the synchrous counter are set very close to the same time, they may not be set at <u>exactly</u> the same time.

This means that there is a rapidly changing transient state of the counter.

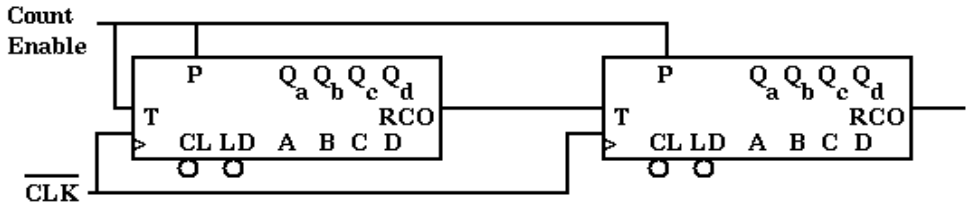
If it passes through all one's it will cause a 'glitch' on the ripple carry out.

You are asked to look for this in Lab 1, but you may not see it!

Care is required of the Ripple Carry Output: It can have glitches: Any of these transition paths are possible!



To cascade synchronous counters (to count more bits):
P is "count Enable"
RCO and T are daisy chained

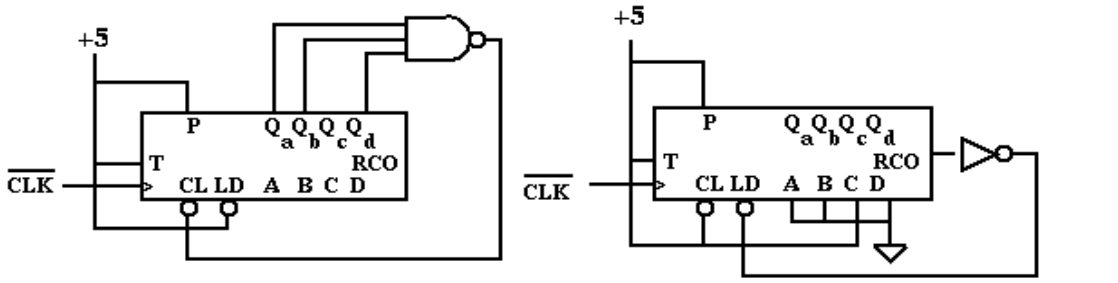


The '163 will 'count' ONLY if P and T are <u>both</u> high

Note that RCO is the AND of all four bits and T.

So if this is input to the T input of the next higher nibble, it indicates that all bits below are set, so the next higher nibble should count.

P is 'count enable', and P and T should be tied together ONLY for the least significant 4 bits of a counter.



This one counts 0,1,2, ... , 11, 0, 1 ... **This one counts 4, 5, ... , 15, 4, 5...**

With a little ingenuity, you can achieve all kinds of count sequences. These are both divide by twelve circuits.