Electric Arithmetic, Flip Flops & The DIGISIM Tool

Review of Objectives

After this lecture, you should be able to……

- **Draw a ripple carry adder block diagram**
- **Use an iterative technique to develop the carry look ahead adder**
- **Review the different types and function of flip flop used as the basic element of sequential logic and finite state machines**
  - **S-R / D type / T and J-K FF's**
  - **Synchronization: Latch, clocked, edge triggered**

Full Adder Truth Table

- $s_i = a_i \oplus b_i \oplus c_i$

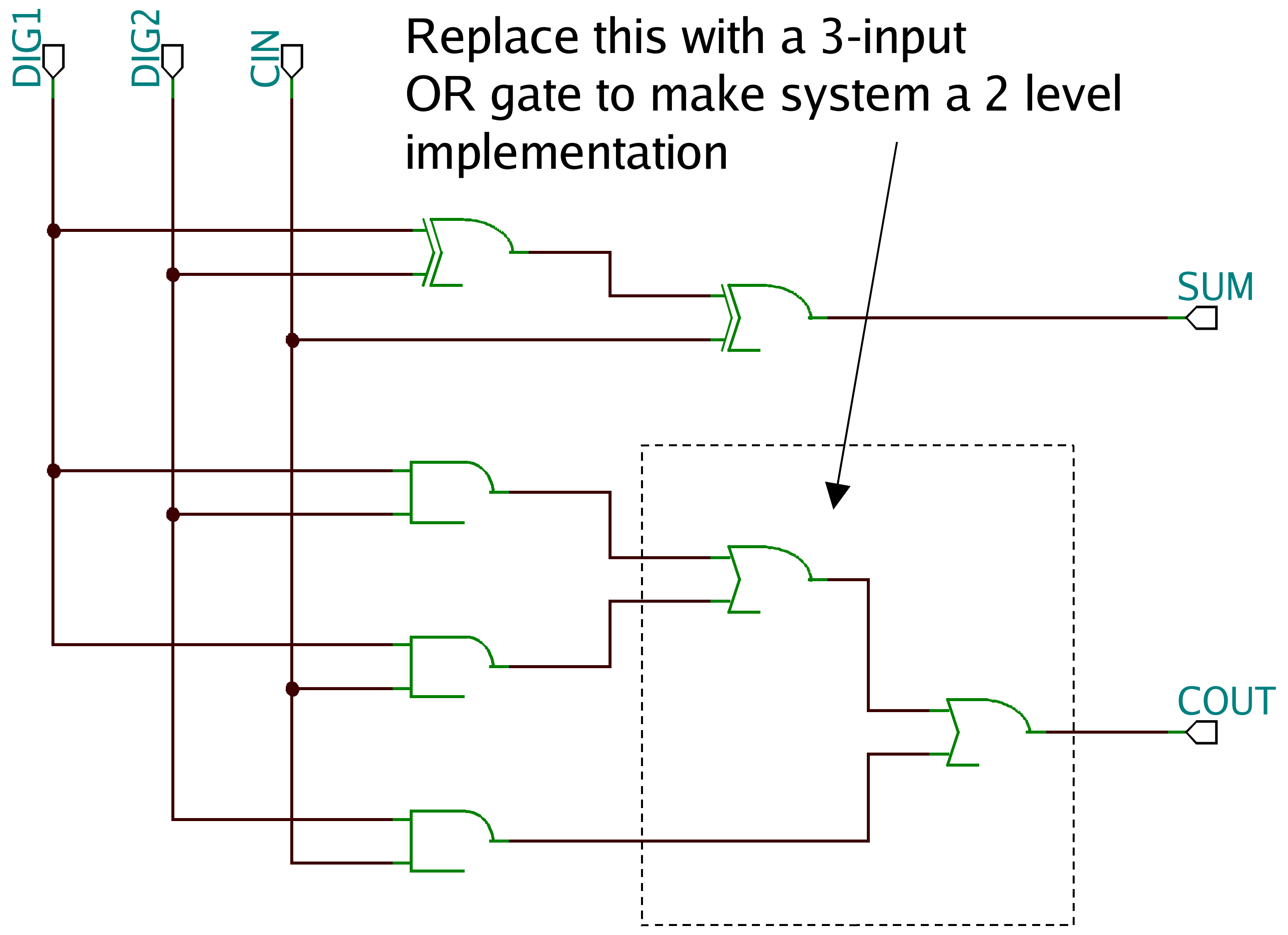
{ XOR'd inputs }

- $c_{i+1} = a_i b_i + a_i c_i + b_i c_i$

{after minimization}

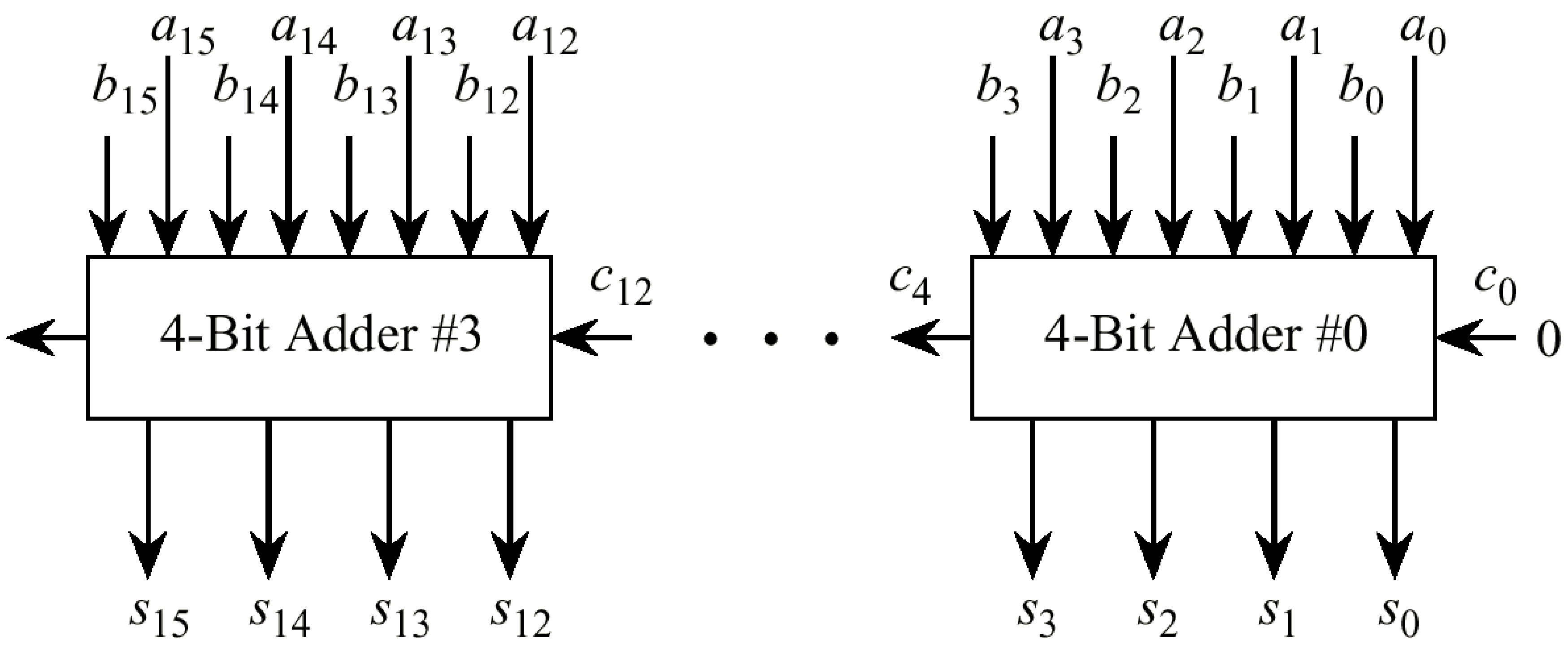
ai	bi	ci	si	ci+1
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

Full Adder Circuit Using 2-input gates


DIG1
DIG2
CIN
Replace this with a 3-input OR gate to make system a 2 level implementation
SUM
COUT

Constructing Larger Adders

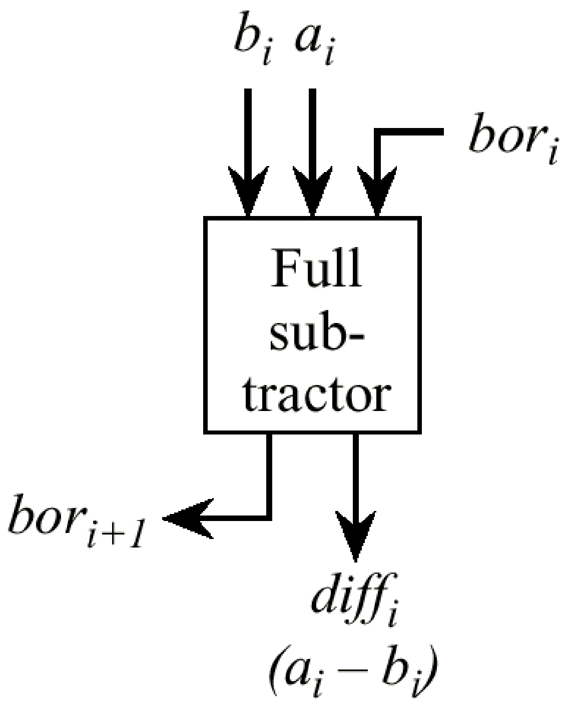
- **A 16-bit adder can be made up of a cascade of four 4-bit ripple-carry adders. Circuit gets slower as it gets bigger…..**



Full Subtractor

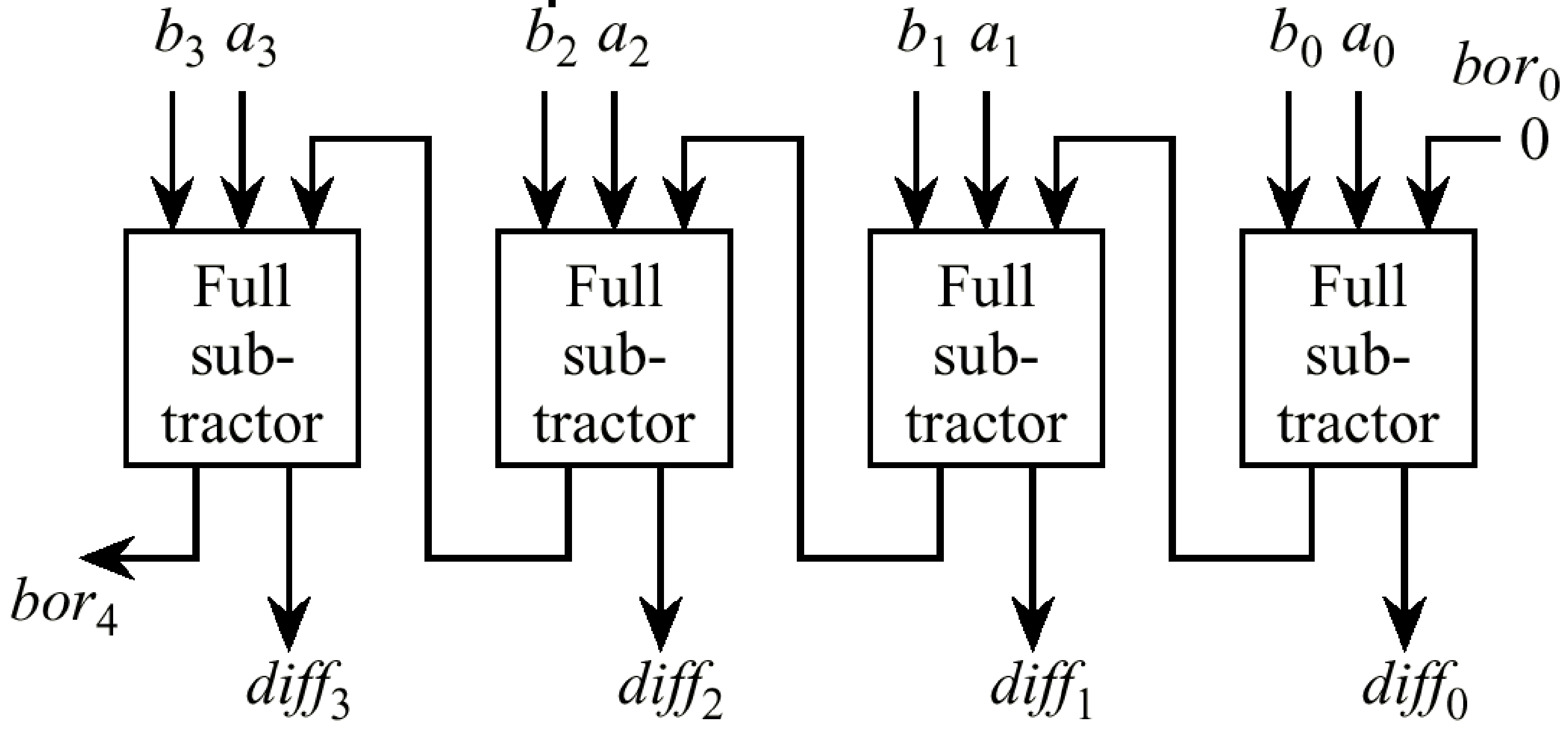
- **Truth table and schematic symbol for a ripple-borrow subtractor:**

a_i	b_i	bor_i	$diff_i$	bor_{i+1}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1



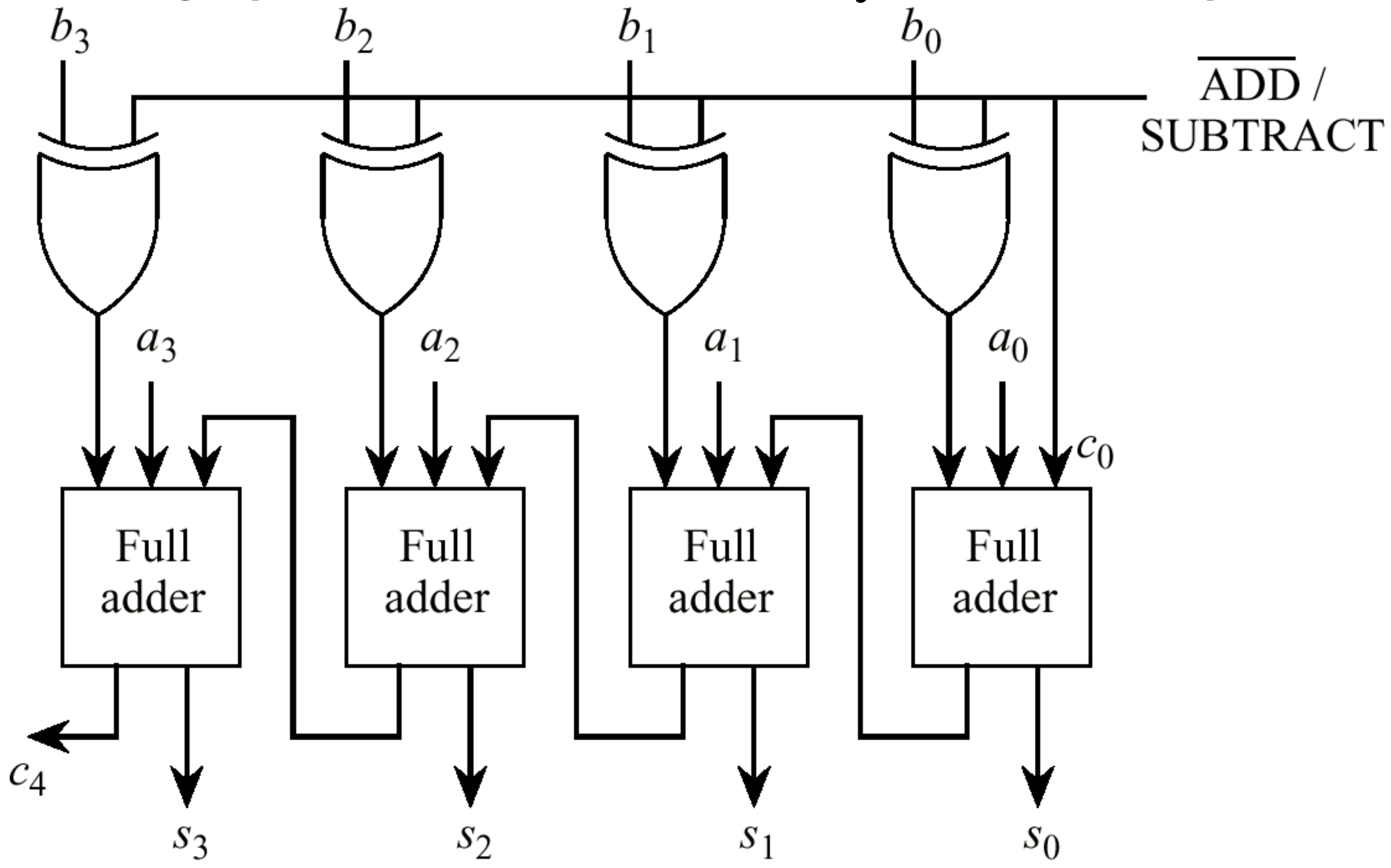
Ripple-Borrow Subtractor

- **A ripple-borrow subtractor can be composed of a cascade of full subtractors.**
- **Two binary numbers *A* and *B* are subtracted from right to left, creating a difference and a borrow at the outputs of each full subtractor for each bit position.**



Combined Adder/Subtractor

- **A single ripple-carry adder can perform both addition and subtraction, by forming the two's complement negative for *B* when subtracting. (Note that +1 is added at c_0 for two's complement.)**



Carry-Lookahead Addition

$$s_i = \overline{a_i}\,\overline{b_i}c_i + \overline{a_i}b_i\overline{c_i} + a_i\overline{b_i}\,\overline{c_i} + a_ib_ic_i$$

$$c_{i+1} = b_ic_i + a_ic_i + a_ib_i$$

$$c_{i+1} = a_ib_i + (a_i + b_i)c_i$$

$$c_{i+1} = G_i + P_ic_i$$

$\boldsymbol{G}_i = a_ib_i$ and $P_i = a_i + b_i$

$\boldsymbol{c}_0 = 0$

$\boldsymbol{c}_1 = G_0$

$\boldsymbol{c}_2 = G_1 + P_1G_0$

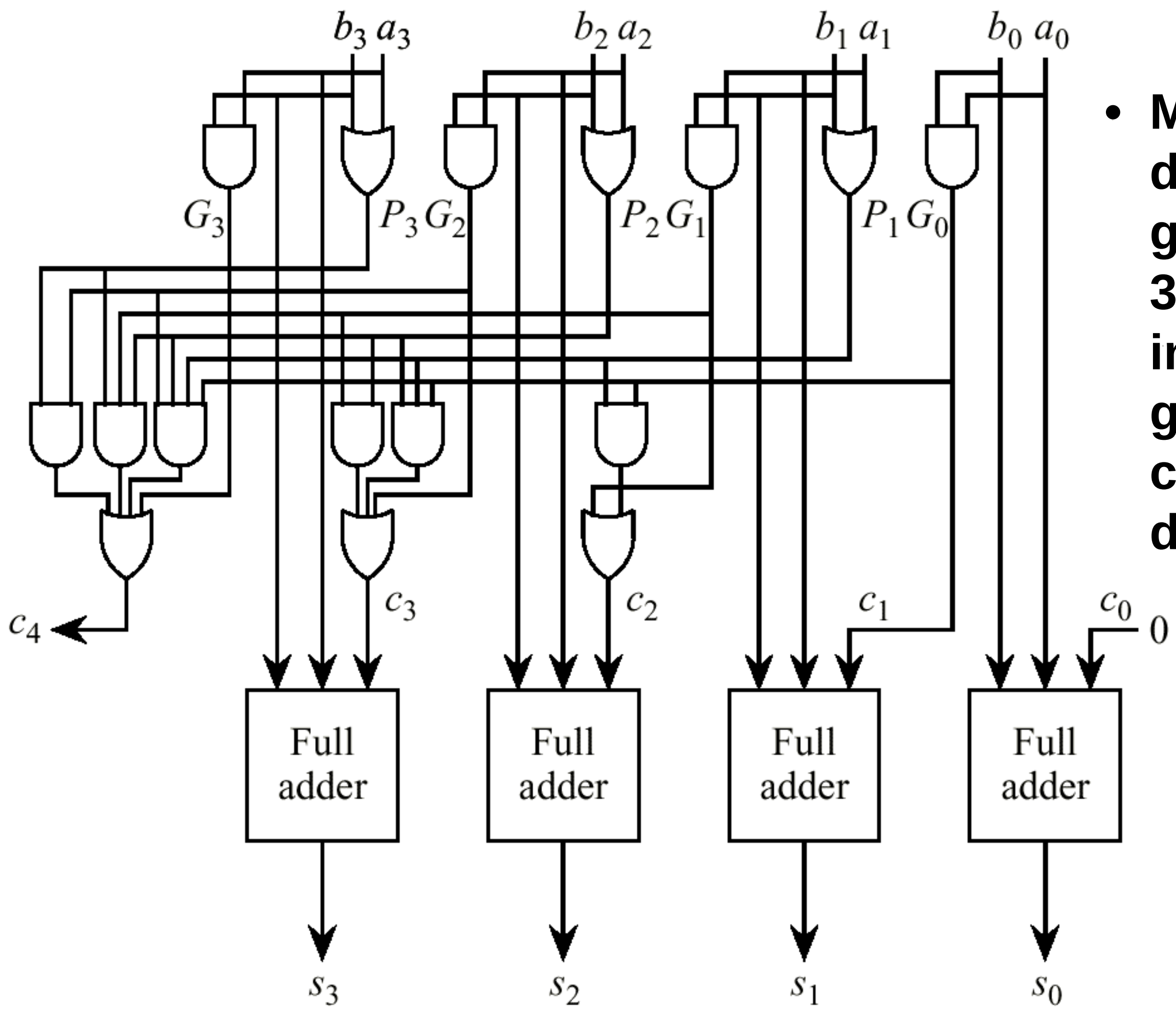
$\boldsymbol{c}_3 = G_2 + P_2G_1 + P_2P_1G_0$

$\boldsymbol{c}_4 = G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0$

- **Carries are represented in terms of G_i (generate) and P_i (propagate) expressions.**

Notice that carry terms are generated from inputs and not other carry terms

Carry Lookahead Adder

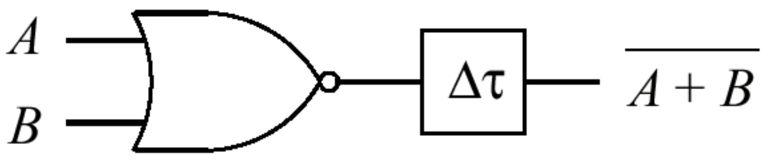


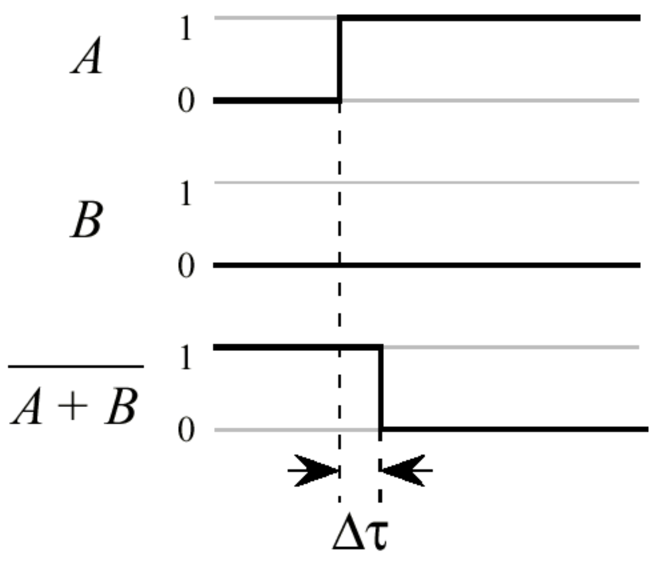
- Maximum gate delay for the carry generation is only 3. The full adders introduce two more gate delays. Worst case path is 5 gate delays.

The Sequential Logic Building Block

The Flip Flop

NOR Gate with Lumped Delay



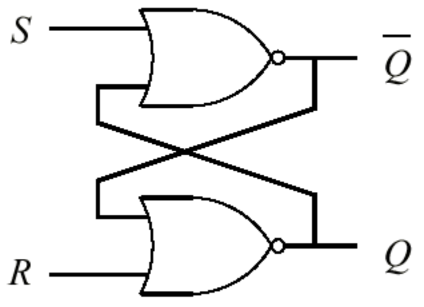


Timing Behavior

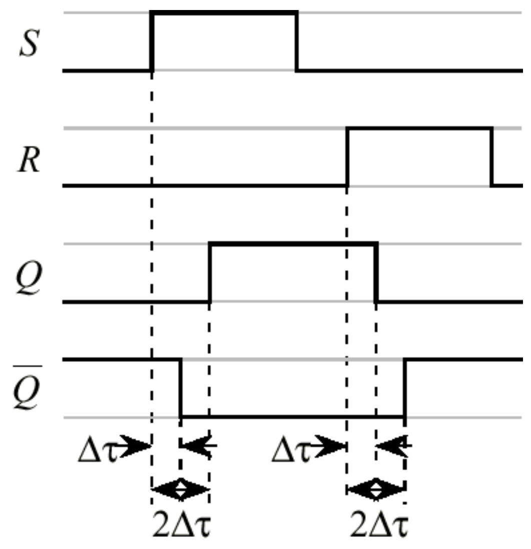
- **The delay between input and output (which is lumped at the output for the purpose of analysis) is at the basis of the functioning of an important memory element, the *flip-flop*.**

S-R Flip-Flop

- The S-R flip-flop is an active high (positive logic) device.

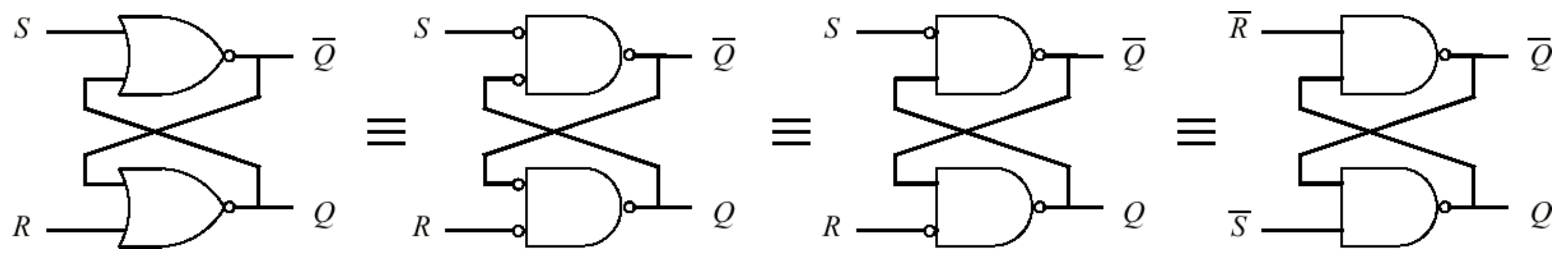


Q_t	S_t	R_t	Q_{t+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	(disallowed)
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	(disallowed)

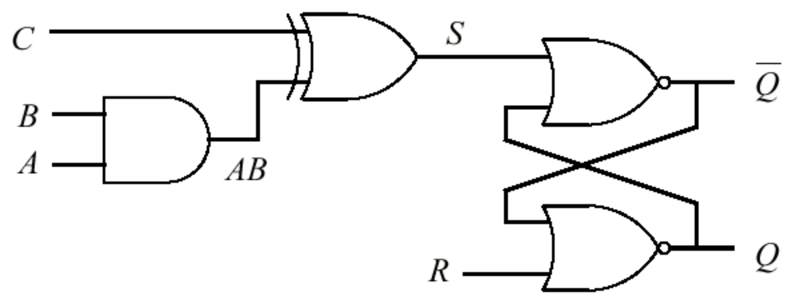


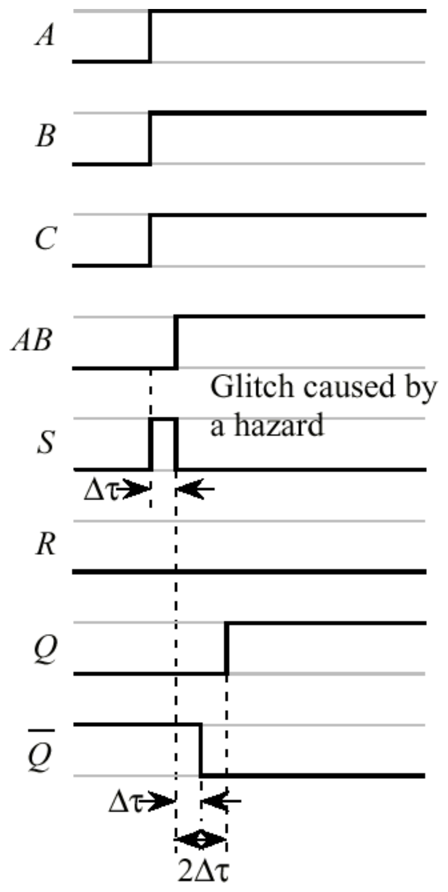
Timing Behavior

NAND Implementation of S-R Flip-Flop



A Hazard

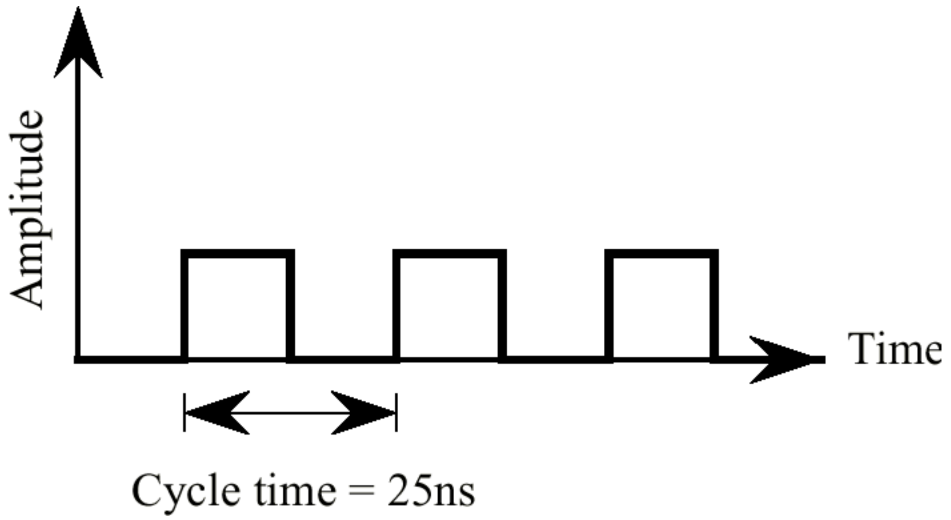




Timing Behavior

• It is desirable to be able to “turn off” the flip-flop so it does not respond to such hazards.

A Clock Waveform: The Clock Paces the System



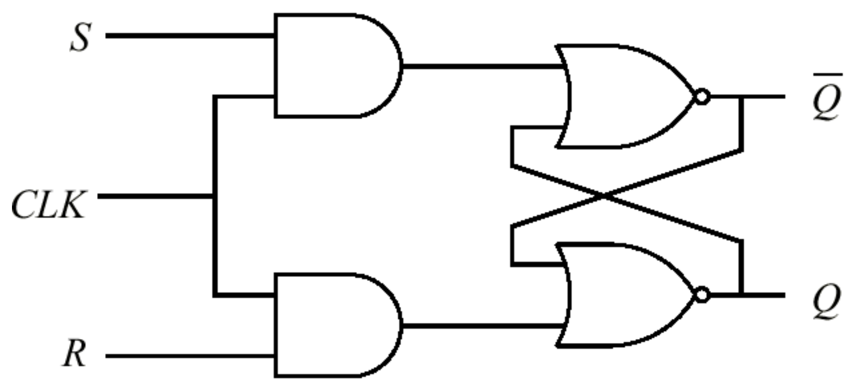
• In a positive logic system, the "action" happens when the clock is high, or positive. The low part of the clock cycle allows propagation between subcircuits, so their inputs settle at the correct value when the clock next goes high.

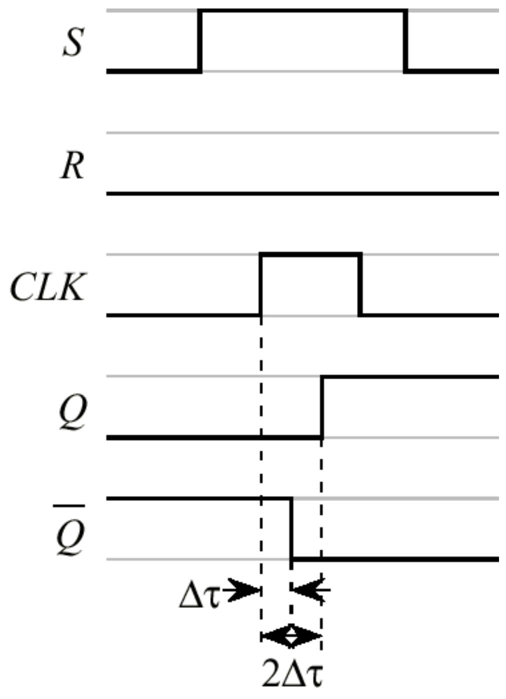
Scientific Prefixes

- For computer memory, 1K = 2^{10} = 1024. For everything else, like clock speeds, 1K = 1000, and likewise for 1M, 1G, *etc.*

Prefix	Abbrev.	Quantity	Prefix	Abbrev.	Quantity
milli	m	10^{-3}	Kilo	K	10^{3}
micro	μ	10^{-6}	Mega	M	10^{6}
nano	n	10^{-9}	Giga	G	10^{9}
pico	p	10^{-12}	Tera	T	10^{12}
femto	f	10^{-15}	Peta	P	10^{15}
atto	a	10^{-18}	Exa	E	10^{18}

Clocked S-R Flip-Flop

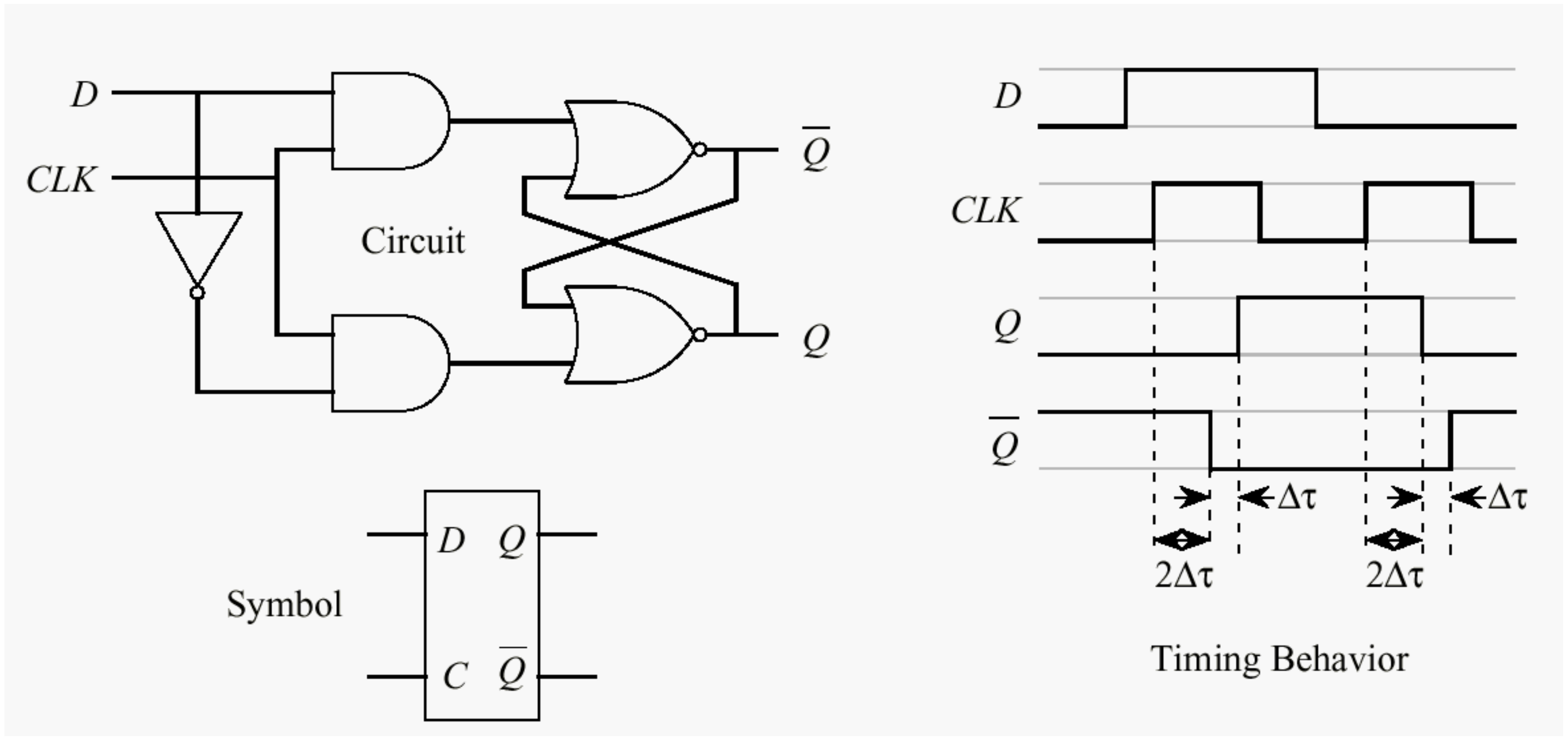




Timing Behavior

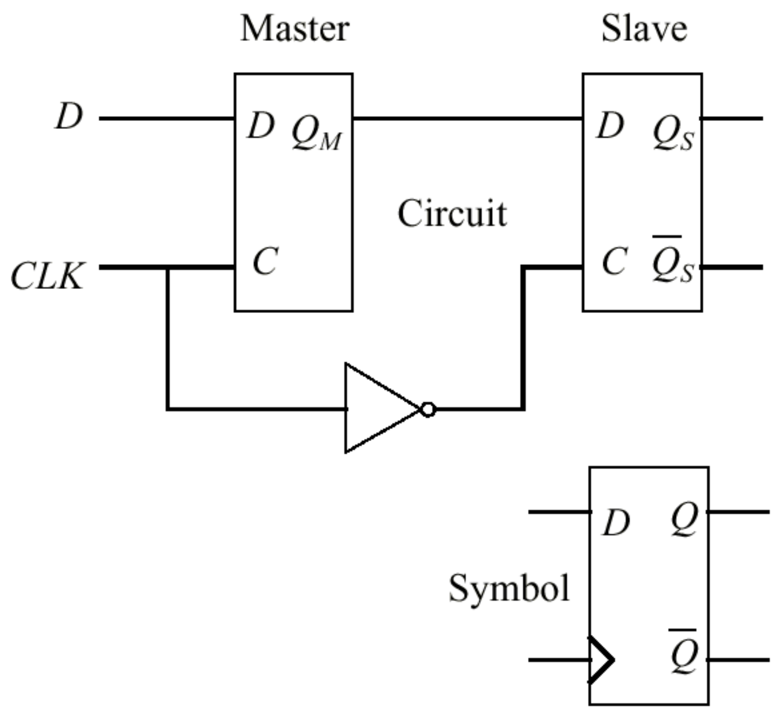
- **The clock signal, CLK, enables the S and R inputs to the flip-flop.**

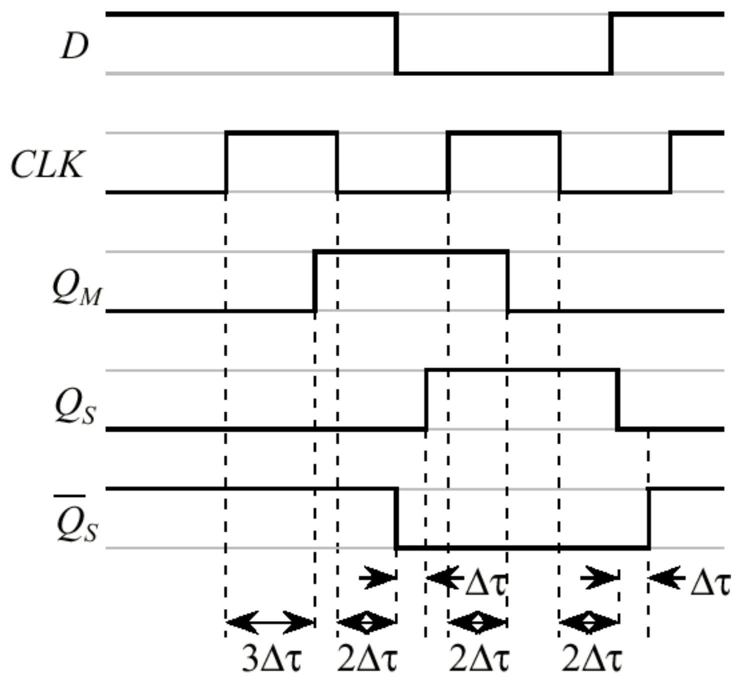
Clocked D Flip-Flop



• The clocked D flip-flop, sometimes called a *latch*, has a potential problem: If D changes while the clock is high, the output will also change. The *Master-Slave* flip-flop (next slide) addresses this problem.

Master-Slave Flip-Flop





Timing Behavior

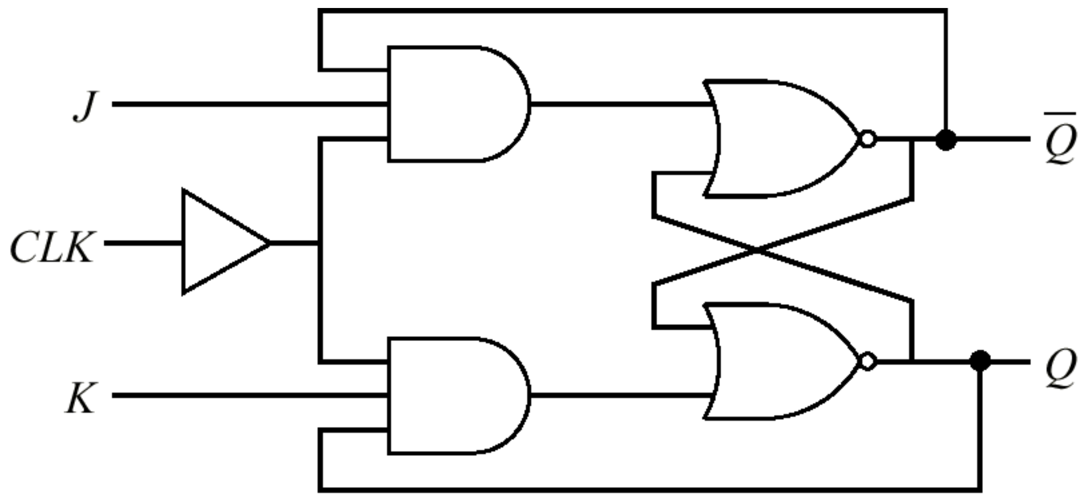
- **The rising edge of the clock loads new data into the master, while the slave continues to hold previous data. The falling edge of the clock loads the new master data into the slave.**

Clocked J-K Flip-Flop

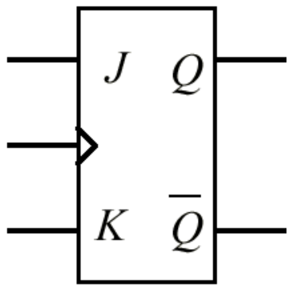
• The J-K flip-flop eliminates the disallowed S=R=1 problem of the S-R flip-flop, because Q enables J while Q' disables K, and vice-versa.

• However, there is still a problem. If J goes momentarily to 1 and then back to 0 while the flip-flop is active and in the reset state, the flip-flop will "catch" the 1. This is referred to as "1's catching."

• The J-K Master-Slave flip-flop (next slide) addresses this problem.

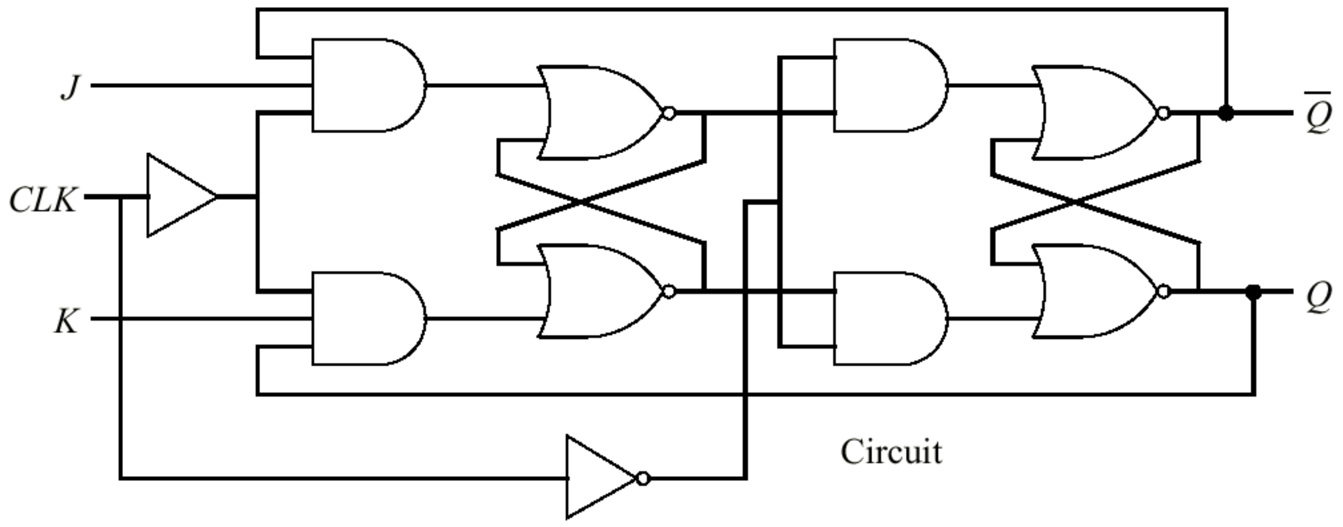


Circuit

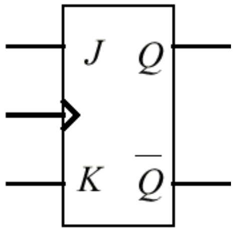


Symbol

Master-Slave J-K Flip-Flop

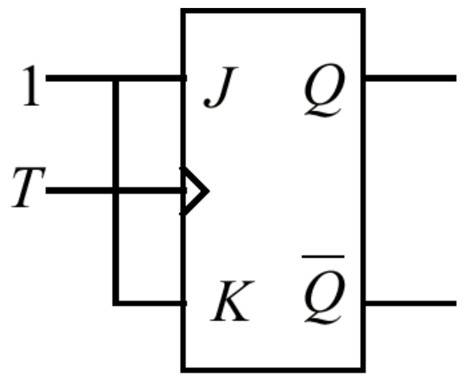


Circuit

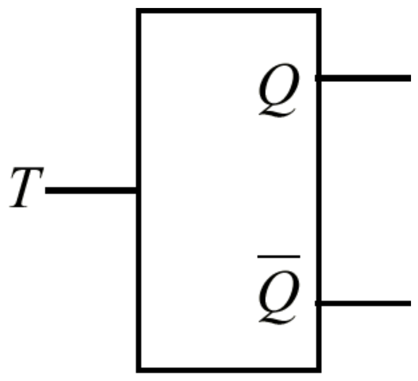


Symbol

Clocked T Flip-Flop



Circuit

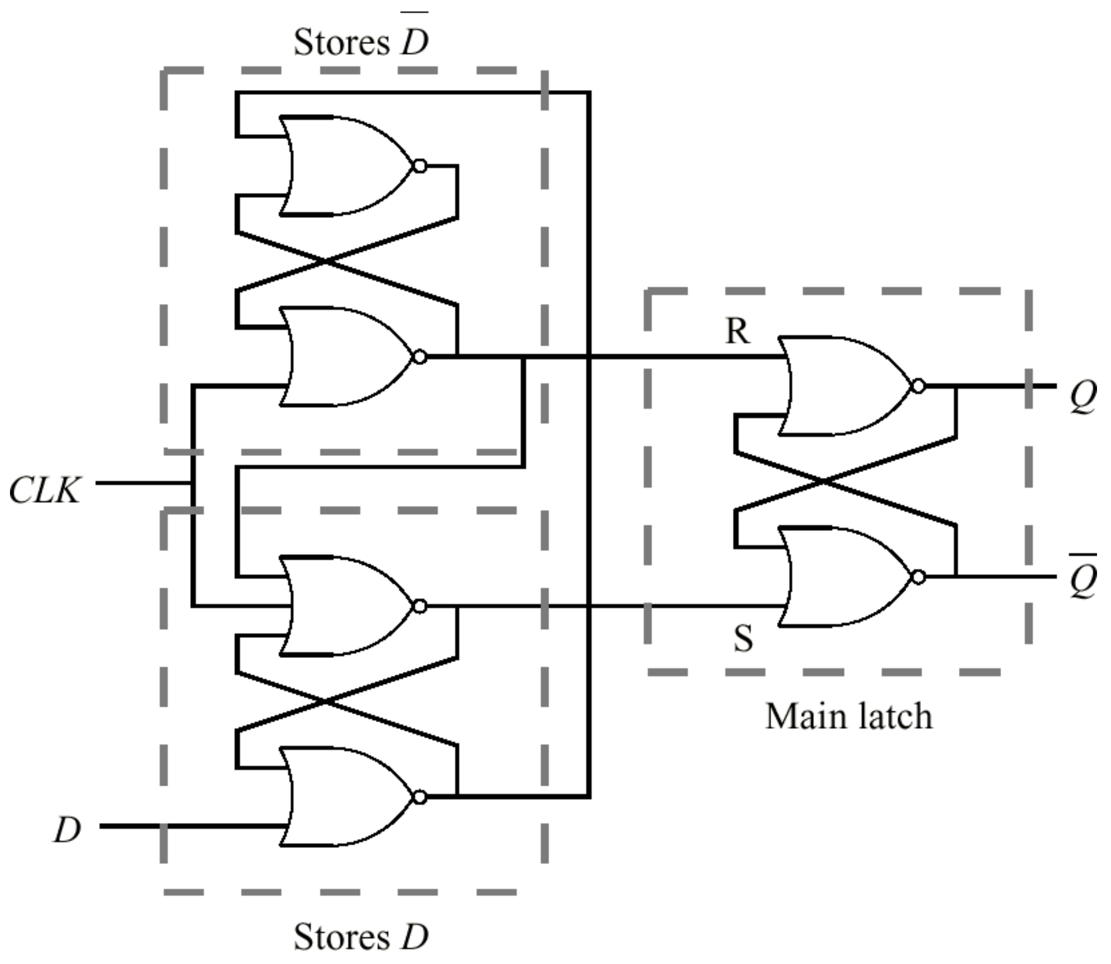


Symbol

- The presence of a constant 1 at J and K means that the flip-flop will change its state from 0 to 1 or 1 to 0 each time it is clocked by the T (Toggle) input.

Negative Edge-Triggered D Flip-Flop

- When the clock is high, the two input latches output 0, so the Main latch remains in its previous state, regardless of changes in D.
- When the clock goes high-to-low, values in the two input latches will affect the state of the Main latch.
- While the clock is low, D cannot affect the Main latch.



Review of Objectives

We…….

- **Constructed a ripple carry adder block diagram**
- **Developed a faster system called the carry look ahead adder**
- **Examined the different types and function of flip flop used as the basic element of sequential logic and finite state machines**
  - **S-R / D type / T and J-K FF's**
  - **Synchronization: Latch, clocked, edge triggered**

Digisim

- This can be run on your own PC or through the gl server
- It is a relatively easy tool to use for logic simulation
- Timing diagrams can be generated for sychronous circuit and FSM (finite state machine) validation
- A quick demo follows......