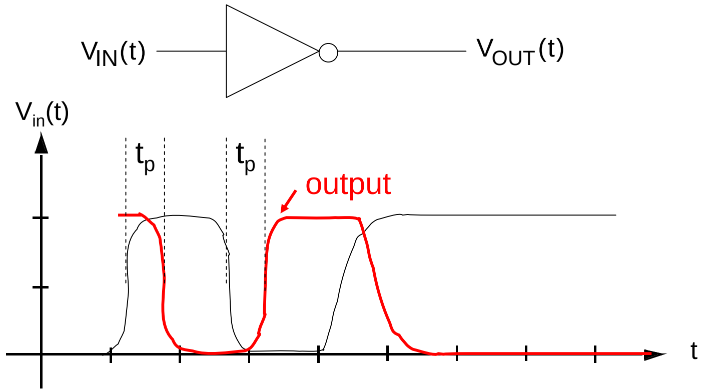
More Digital Logic

- Gate delay and signal propagation
- Clocked circuit elements (flip-flop)
- Writing a word to memory
- Simplifying digital circuits: Karnaugh maps

PROPAGATION DELAY

The propagation delay t_p is defined as:

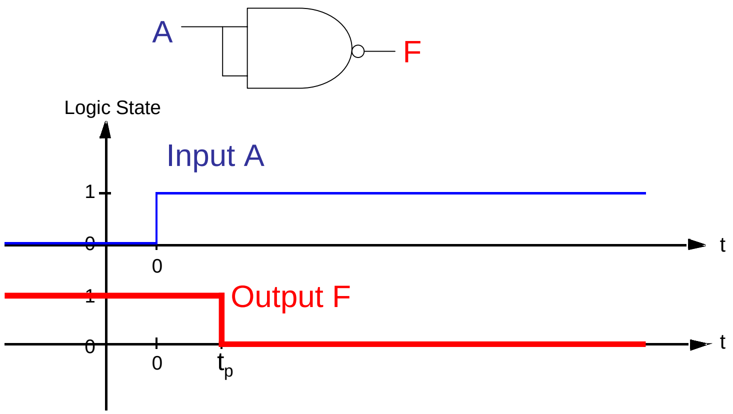
t_p = time when output is halfway between initial and final value   -
    time when input is halfway between initial and final value.



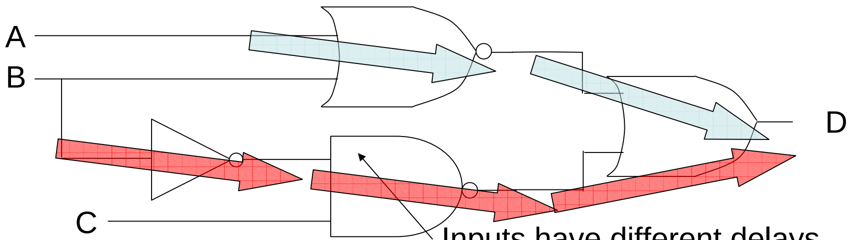
Low-to-high and high-to-low transitions could have different t_p.

PROPAGATION DELAY

To get an approximate idea of the effects of delay, we make the transitions look instantaneous (though they are exponential).



PROPAGATION DELAY



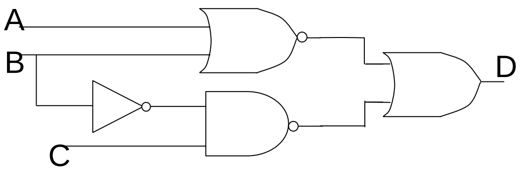
Inputs have different delays, but we ascribe a single worst-case delay t_p to every gate

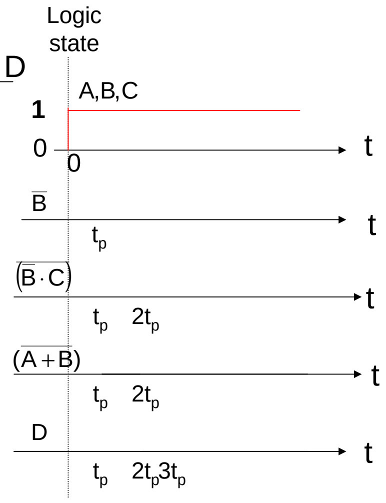
How many gate delays for shortest path? ANSWER : 2

How many gate delays for longest path? ANSWER : 3

TIMING DIAGRAMS

Different delays through different paths can create “false” output: Circuit computes using partially updated signals.





Note $\overline{B}$ becomes valid one gate delay after B switches

Note that $(\overline{\overline{B}\cdot C})$ becomes valid two gate delays after B&C switch, because the invert function takes one delay and the NAND function a second.

The final OR gate creates one more delay.

SYNCHRONOUS LOGIC

We have now seen that a circuit can produce nonsensical output due to differing delay paths in the circuit.

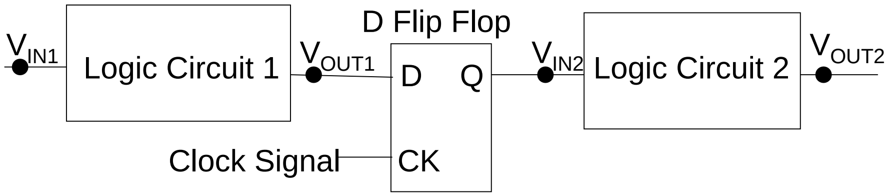
Presumably, the output of a logic circuit might serve as the input of a second logic circuit.

How do we prevent the second circuit from using and passing on this false information?

Answer: Include “gatekeeper” components that pass on data only when enough time has passed to guarantee validity

Clocked (Synchronous) components: flip-flops

SYNCHRONOUS CIRCUIT



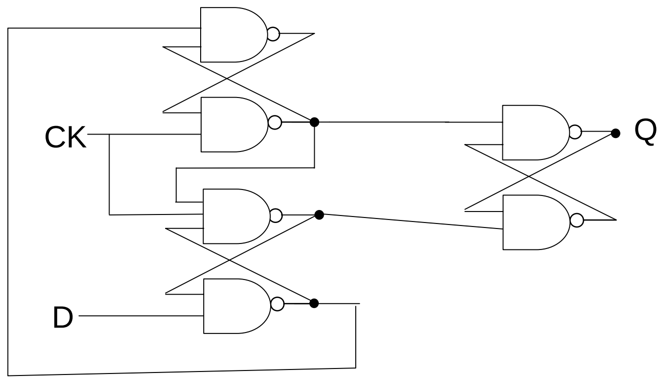
The D Flip Flop is a synchronous (clocked) sequential (memory) circuit.

At the instant the clock signal CK rises from logic 0 to logic 1, the output Q is set equal to the input D.

At all other times, the output Q remains the same.

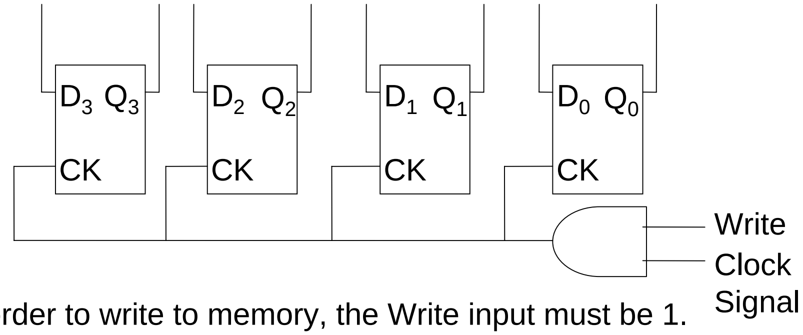
The flip flop prevents Logic Circuit 2 from receiving a new input value, until the clock transition allows the data to pass through.

FLIP FLOP CIRCUIT DIAGRAM



We may cover digital circuits with feedback later in the course.

MAKING MEMORIES



- In order to write to memory, the Write input must be 1.
- Note we can only write to all four cells at once!
- To change only 1 bit, e. g., change D_2 to 1, set
  D3 = Q3 D2 = 1 D1 = Q1 D0 = Q0
- To change 1 bit on CalBot board using software interface, use **bitwise OR** “**|**” with a **mask**: Q = Q **|** 0100
  (where Q is defined as $Q_3Q_2Q_1Q_0$)

- ❖ How do you set one bit to logic 0 using the CalBot interface?

CREATING A BETTER CIRCUIT

What makes a better digital circuit? Fast and low cost = better.

•Fewer stages

•Fewer total number of individual gates

•Fewer types of gates

•Fewer inputs on each gate (multi-input gates are slower)

In general, simplifying a digital circuit to minimize the number of gates is computationally intractable (uses very large amount of time and space, worse than NP-hard)

The method of Karnaugh maps reduces the number of inputs per gate.

- ❖ It is interesting to study the computational effort needed to analyze and simplify digital circuits. Some interesting results (for those who know about complexity, maybe discussed at end of semester):
- ❖ The problem of deciding whether there is a combination of inputs to a digital circuit that will generate an output of 1 is in a special class of problems called NP-complete. If you are smart enough to figure out a way to solve this problem in polynomial time, then the world will be able to solve all sorts of hard problems in polynomial time and you will win a Fields Medal.
- ❖ The number of gates in a circuit is mathematically related to the time complexity of the problem it decides. If you can find an NP problem that cannot be decided with a circuit that has polynomial number of gates, then you have proved that some NP problems cannot be decided in polynomial time and you win a Fields Medal.

KARNAUGH MAPS

To find a simpler sum-of-products expression,

1. Write the truth table of your circuit into a special table.

		B	
		0	1
A	0		
	1		

2 Inputs

		BC			
		00	01	11	10
A	0				
	1				

3 Inputs

		CD			
		00	01	11	10
AB	00				
	01				
	11				
	10				

4 Inputs

2. For each “1”, circle the biggest 2m by 2n block that includes that “1”.
3. Write the product that corresponds to that block.

- ❖ This method simplifies circuits to smaller sum-of-products (ANDs and ORs). Could you use these maps to simplify with XOR?

EXAMPLE: ADDER

A	B	C	S_1	S_0
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Input **Output**

Simplification for S_1:

		BC			
		00	**01**	**11**	**10**
A	**0**				
	1				

- ❖ Use the Karnaugh map method to simplify S_0. Is there a simpler circuit for S_0?