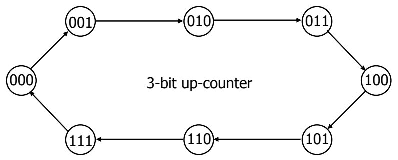
Lecture 17

- Logistics
  - Lab 7 this week
  - HW6 is due Friday
  - Office Hours
    - Mine: Friday 10:00-11:00 as usual
    - Sara: Thursday 2:30-3:20 CSE 220
    - Josh: Thursday 3:30-4:20 CSE 002
  - Midterm delayed until next Wednesday
    - Will cover material up to Friday's lecture
- Last two lectures
  - Registers, Counters, Counter Finite State Machines (FSM)
  - Sequential Verilog
- Today
  - Another counter FSM
  - Timing issues
    - Timing terminology and issues and solutions (e.g. clock skew)
    - Asynchronous inputs and issues and solutions (e.g. debouncing)

Another 3-bit up counter: now with T flip flops

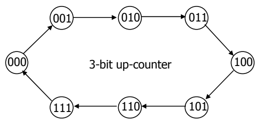
1. Draw a state diagram
2. Draw a state-transition table
3. Encode the next-state functions
   - Minimize the logic using K-maps
4. Implement the design

1. Draw a state diagram



2. Draw a state-transition table

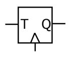
- Like a truth-table
  - State encoding is easy for counters → Use count value



	current state	next state	
0	000	001	1
1	001	010	2
2	010	011	3
3	011	100	4
4	100	101	5
5	101	110	6
6	110	111	7
7	111	000	0

3. Encode the next state functions

T flip-flops



Ti = 1 iff Ni ≠ Ci

T0 := 1
T1 := C0
T2 := C0 C1

C2	C1	C0	N2	N1	N0	T2	T1	T0
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1	1
1	1	0	1	1	1	0	0	1
1	1	1	0	0	0	1	1	1

T0

			C2	C2
	1	1	1	1
C0	1	1	1	1
		C1	C1	

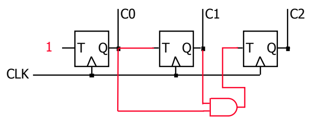
T1

			C2	C2
	0	0	0	0
C0	1	1	1	1
		C1	C1	

T2

			C2	C2
	0	0	0	0
C0	0	1	1	0
		C1	C1	

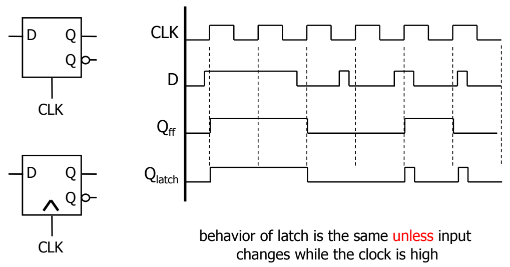
4. Implement the design



The “WHY” slide

- Timing issues
  - For sequential logic, “timing” is critical because for the same inputs, the output could be different at different times (like T-flip flops). In order to achieve desired outputs, timing has to be taken into consideration.
  - Transistors, chips, and even wires have their own delays. Because of this, nothing could ever be perfectly synchronized. It is important to understand how fast a clock can tick based on these delays and what the common issues are in making computers to run fast and accurately.
  - There are synchronous and asynchronous inputs. For example, typing on the keyboard, you are putting in asynchronous inputs to the computer. Asynchronous inputs can change the outputs immediately regardless of the clock state, and it is important to know how to handle that.

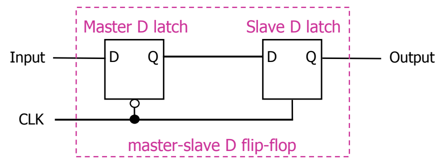
Latches versus flip-flops



behavior of latch is the same unless input changes while the clock is high

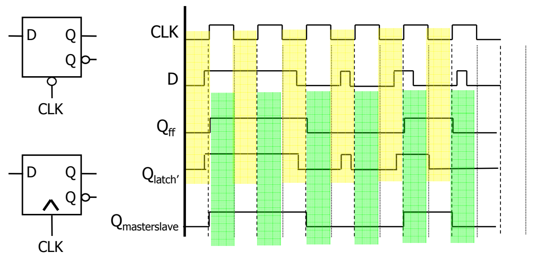
For most applications, it is not good to see Input changes instantaneously at the output

The master-slave D (polarity reversed from previous class)



Because of the timing issue, it was good to use two latches as master-slave configuration or use one flip-flop.

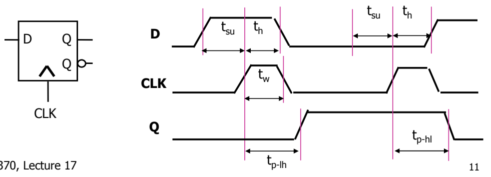
Master-Slave D implements D flip-flop



For most applications, it is not good to see Input changes instantaneously at the output

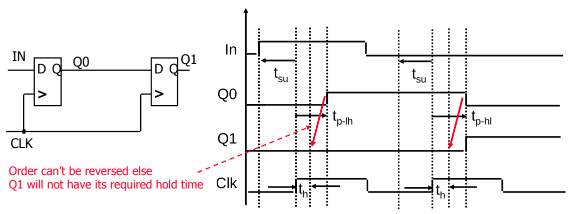
Timing terminology and constraints for a FF

- Setup time t_{su}: Amount of time the input must be stable before the clock transitions high (or low for negative-edge triggered FF)
- Hold time t_h: Amount of time the input must be stable after the clock transitions high (or low for negative-edge triggered FF)
- Clock width t_w : Minimum clock width that must be met in order for FF to work properly
- Propagation delays $t_{p\text{-}lh}$ and $t_{p\text{-}hl}$: Propagation delay (high to low, low to high) (longer than hold time)



Cascading flip-flops

- Flip-flop propagation delays exceed hold times
  - Second stage commits its input before Q0 changes

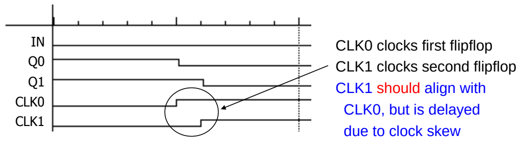


$t_{p\text{-}hl}$, $t_{p\text{-}lh} > t_h$

$t_p + t_{su} < t_{cycle}$

Side note: Clock skew

- Goal: Clock all flip-flops at the same time
  - Difficult to achieve in high-speed systems
    - Clock delays (wire, buffers) are comparable to logic delays
  - Problem is called clock skew



Original state: IN = 0, Q0 = 1, Q1 = 1
Next state: Q0 = 0, Q1 = 0 (should be Q1 = 1)

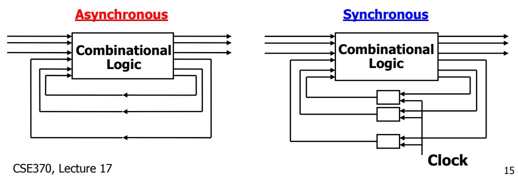
  - Avoiding clock skew: design identical delays

System considerations

- Use edge-triggered flip-flops wherever possible
  - Avoid latches
  - Most common: Master-slave D
- Basic rules for correct timing
  - Clock flip-flops synchronously (all at the same time)
    - No flip-flop changes state more than once per clock cycle
    - FF propagation delay > hold time
  - Avoid mixing positive-edge triggered and negative-edge triggered flip-flops in the same circuit

Asynchronous versus synchronous

- Asynchronous
  - State changes occur when state inputs change
  - Feedback elements may be wires or delays
- Synchronous
  - State changes occur synchronously
  - Feedback elements are clocked

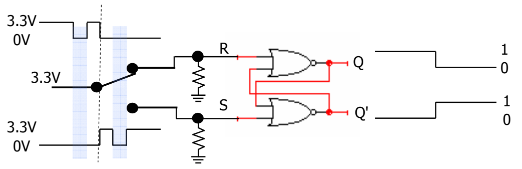


Asynchronous inputs

- Clocked circuits are synchronous
  - Circuit changes state only at clock edges
  - Signals (voltages) settle in-between clock edges
- Unclocked circuits or signals are asynchronous
  - No master clock
  - Real-world inputs (e.g. a keypress) are asynchronous
- Synchronous circuits have asynchronous inputs
  - Reset signal, memory wait, user input, etc.
  - Inputs "bounce"
  - Inputs can change at any time
    - We must synchronize the input to our clock
    - Inputs will violate flip-flop setup/hold times

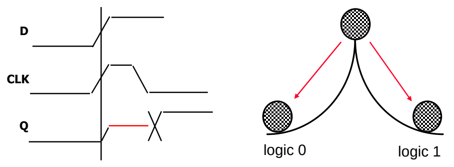
Debouncing

- Switch inputs bounce
  - i. e. don't make clean transitions
- Can use RS latch for debouncing
  - Eliminates dynamic hazards
  - "Cleans-up" inputs



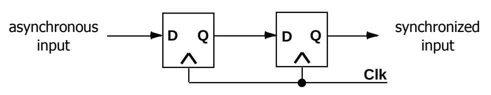
Synchronizer failure

- Occurs when FF input changes near clock edge
  - Input is neither 1 or 0 when clock goes high
  - Output may be neither 0 or 1
    - May stay undefined for a long time
  - Undefined state is called metastability



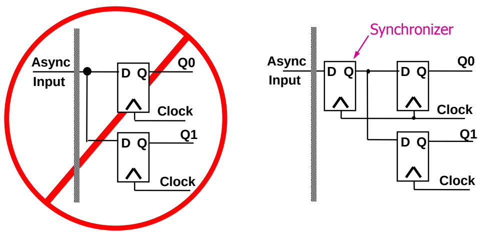
Minimizing synchronizer failures

- Failure probability can never be zero
  - Cascade two (or more) flip-flops
    - Effectively synchronizes twice
    - Both would have to fail for system to fail



Handling asynchronous inputs

- Never fan-out asynchronous inputs
  - Synchronize at circuit boundary
  - Fan-out synchronized signal



Summary: Timing issues with asynchronous inputs

- For sequential logic circuits, timing issues have to be considered.
- Inputs are often asynchronous and can cause problems.
- Different amount of delay at different part of the circuit can cause problems also.
- Solutions:
  - Cascade flip flops in series
  - Incorporate RS latch for debouncing
  - Design to keep timing alignment in mind (length of wires, etc)