EECS 150 - Components and Design Techniques for Digital Systems

Lec 12 - Timing

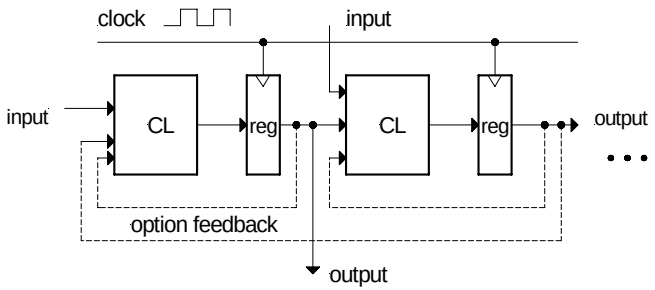
David Culler
Electrical Engineering and Computer Sciences
University of California, Berkeley

http://www.eecs.berkeley.edu/~culler
http://www-inst.eecs.berkeley.edu/~cs150

Outline

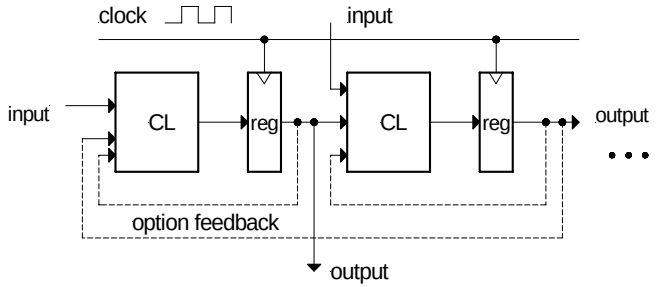
- Performance Limits of Synchronous Systems
- Delay in logic gates
- Delay in wires
- Delay in combinational networks
- Clock Skew
- Delay in flip-flops
- Glitches

Recall: General Model of Synchronous Circuit



- All wires, except clock, may be multiple bits wide.
- Registers (reg)
  - collections of flip-flops
- clock
  - distributed to all flip-flops
  - typical rate?
- Combinational Logic Blocks (CL)
  - no internal state
  - output only a function of inputs
- Particular inputs/outputs are optional
- Optional Feedback

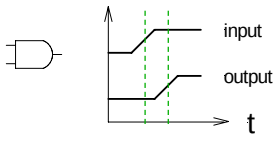
General Model of Synchronous Circuit



- How do we measure performance?
  - operations/sec?
  - cycles/sec?
- What limits the clock rate?
- What happens as we increase the clock rate?

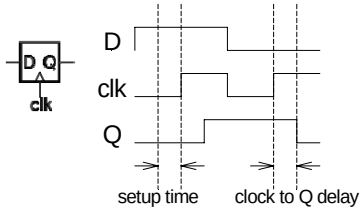
Limitations on Clock Rate

1 **Logic Gate Delay**



- **What are typical delay values?**

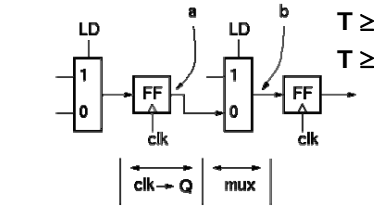
2 **Delays in flip-flops**



- **Both times contribute to limiting the clock period.**

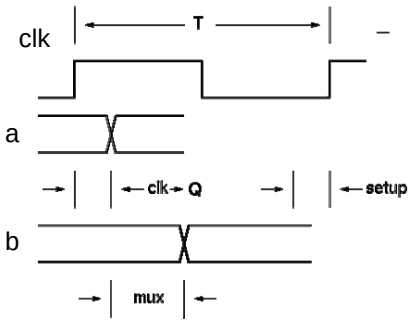
- **What must happen in one clock cycle for correct operation?**
- **Assuming perfect clock distribution (all flip-flops see the clock at the same time):**
  - **All signals must be ready and "setup" before rising edge of clock.**

Example: Parallel-Serial Converter

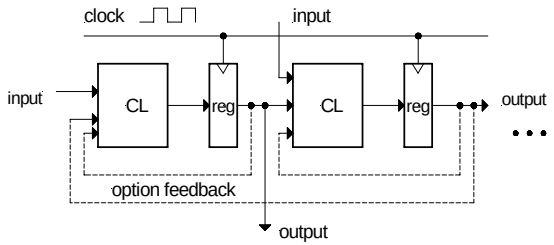


$T \geq$ time(clk→Q) + time(mux) + time(setup)

$T \geq \tau_{clk \to Q} + \tau_{mux} + \tau_{setup}$



General Model of Synchronous Circuit



- **In general, for correct operation:**

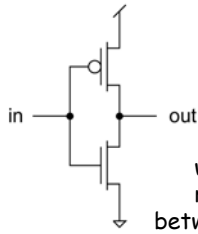
$T \geq$ time(clk→Q) + time(CL) + time(setup)

$T \geq \tau_{clk \to Q} + \tau_{CL} + \tau_{setup}$ **for all paths.**

- **How do we enumerate *all paths*?**
  - **Any circuit input or register output to any register input or circuit output.**
  - **"setup time" for circuit outputs depends on what it connects to**
  - **"clk-Q time" for circuit inputs depends on from where it comes.**

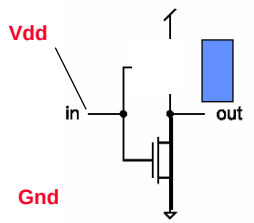
Recall L2: Transistor-level Logic Circuits

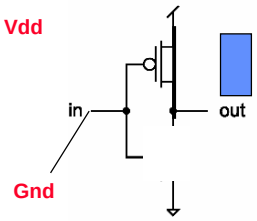
- **Inverter (NOT gate):**



what is the relationship between in and out?

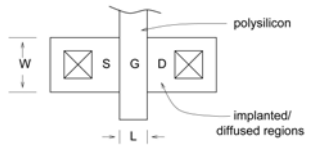
in	out
0 volts	
3 volts	



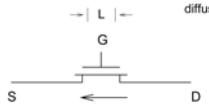


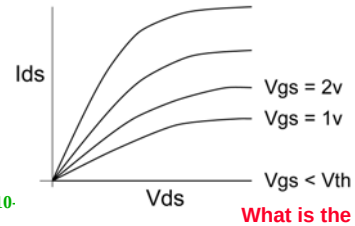
Qualitative Analysis of Logic Delay

- **Improved Transistor Model:**



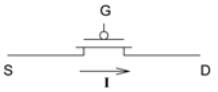
nFET

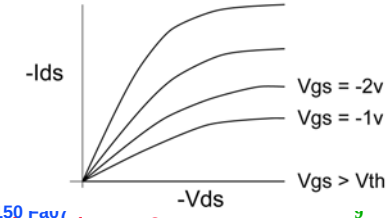




- **We refer to transistor "strength" as the amount of current that flows for a given Vds and Vgs.**
- **The strength is linearly proportional to the ratio of W/L**
  - **Physical property**
- **Turn it on harder allows more current to flow**

pFET

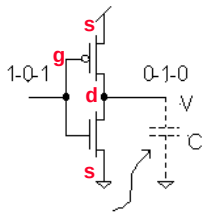




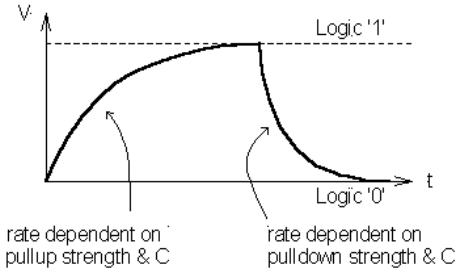
What is the effective resistance?

Gate Switching Behavior

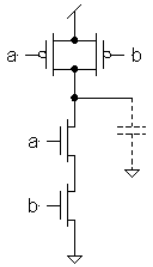
- **Inverter:**

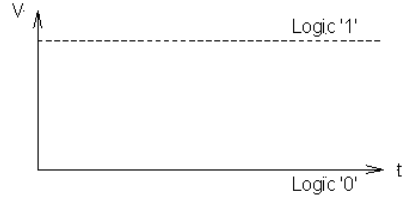


Models inputs to other gates & wire capacitance



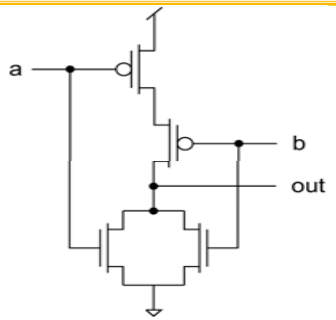
- **NAND gate:**





When does it start? How quickly does it switch?

Clarify your understanding



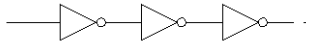
What is the 0 → 1 and 1 → 0 behavior of a NOR gate?

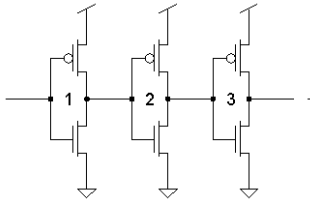
Why do we need pMOS and nMOS devices in a pass gate?

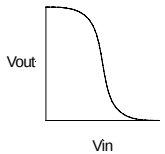
- used for tristate

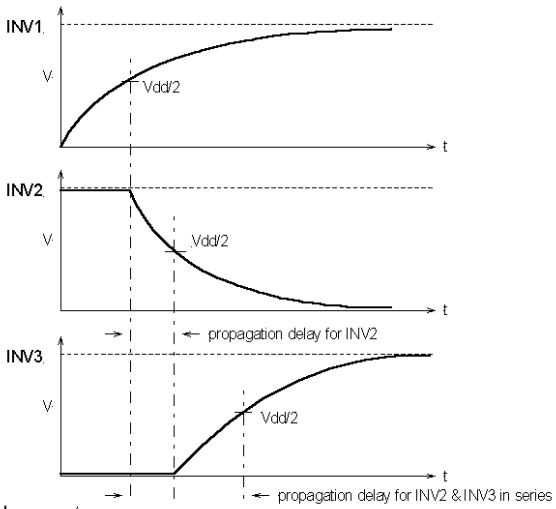
Delays in a series of gates

- **Cascaded gates:**





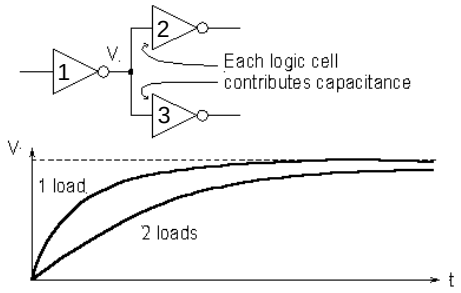




In general:
prop. delay = sum of individual prop. delays of gates in series.

Gate Delay due to fan out

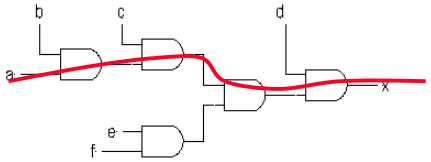
- **Fan-out:**



- **The delay of a gate is proportional to its output capacitance. Because, gates #2 and 3 turn on/off at a later time. (It takes longer for the output of gate #1 to reach the switching threshold of gates #2 and 3 as we add more output capacitance.)**

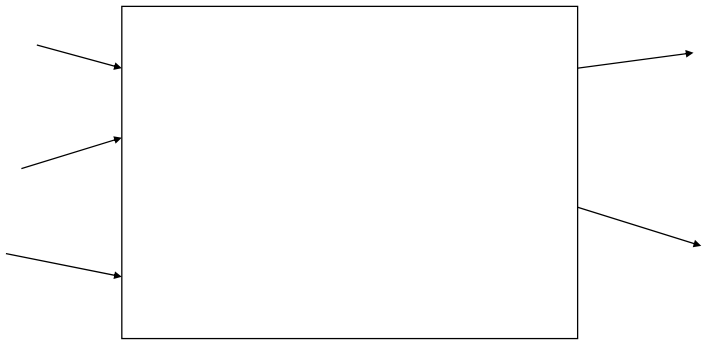
Gate Delay with a general circuit

- **"Fan-in"**
  - **Does it affect the delay of the individual gate?**
  - **When does the gate begin its transition?**



- **What is the delay in this circuit?**
- ***Critical Path:* the path with the maximum delay, from any input to any output.**
  - **In general, we include register set-up and clk-to-Q times in critical path calculation.**
- **Why do we care about the *critical path?***

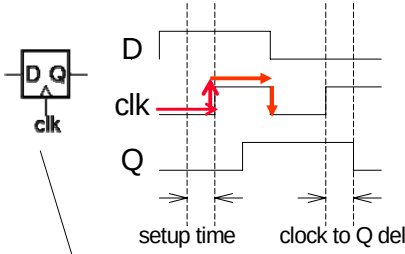
What is the delay through arbitrary combinational logic?



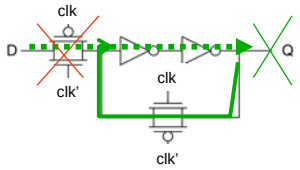
Announcements

- **Reading: K&B 3.5 6.1.5-6.2.3 (were in 9/20 assignment)**
- **K&B 10.6 is great protocol example**
  - **We'll do several of those as we go**
- **HW 5 out today (due 10/12)**
- **Class survey**
- **Lab partners**

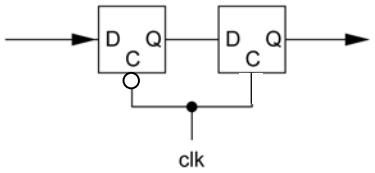
Delay in Flip-flops

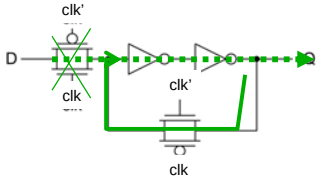


- **Setup time results from delay through *first* latch.**

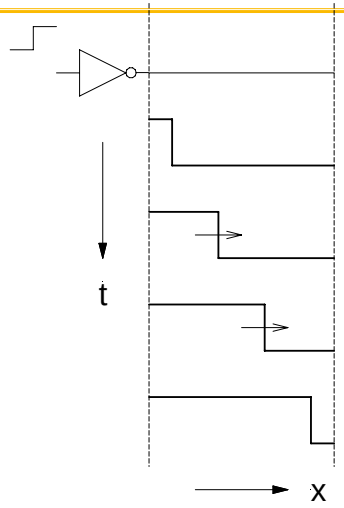


- **Clock to Q delay results from delay through *second* latch.**



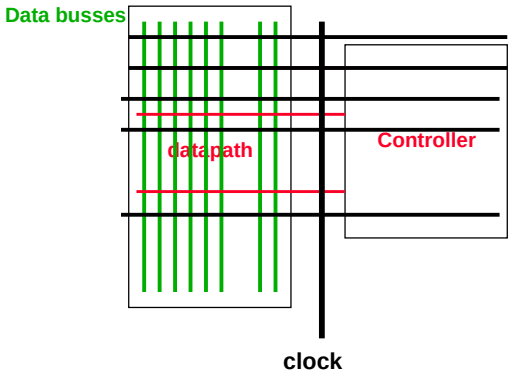


Wire Delay



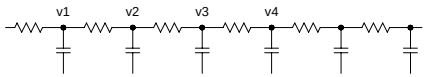
- **In general, wire behave as "transmission lines":**
  - **signal wave-front moves close to the speed of light**
    - **» ~1ft/ns**
  - **Time from source to destination is called the "transit time".**
  - **In ICs most wires are short, and the transit times are relatively short compared to the clock period and can be ignored.**
  - **Not so on PC boards.**
  - **...Or long wires on fast chips**
    - **» Busses**
    - **» Global Control signals**
    - **» Clock**
- **Rule of thumb: wire must be treated as a transmission line if its length exceed λ/100.**

Architectural Level Delay

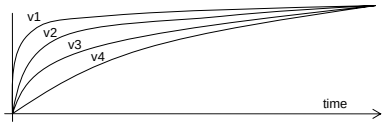


Wire Delay

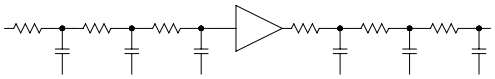
- **Even in those cases where the transmission line effect is negligible:**
  - **Wires posses distributed resistance and capacitance**



  - **Time constant associated with distributed RC is proportional to the *square* of the wire length**



- **For short wires on ICs, resistance is insignificant (relative to effective R of transistors), but C is important.**
  - **Typically around half of C of gate load is in the wires.**
- **For long wires on ICs:**
  - **busses, clock lines, global control signal, etc.**
  - **Resistance is significant, therefore distributed RC effect dominates.**
  - **signals are typically "rebuffered" to reduce delay:**

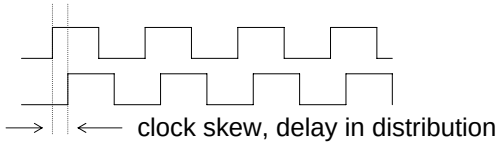


Modern rule of thumb

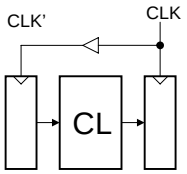
- Transistors are cheap
  - And their local wires
- Wire is what counts
- Often pays to do extra local computation (gates) to reduce wire delay

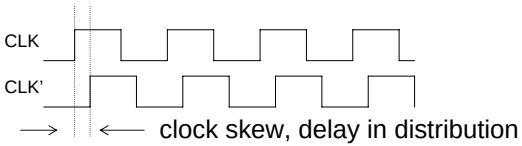
Clock Skew

- Unequal delay in distribution of the clock signal to various parts of a circuit:
  - if not accounted for, can lead to erroneous behavior. (see next)
  - Comes about because:
    - » clock wires have delay,
    - » circuit is designed with a different number of clock buffers from the clock source to the various clock loads, or
    - » buffers have unequal delay.
  - All synchronous circuits experience some clock skew:
    - » more of an issue for high-performance designs operating with very little extra time per clock cycle.



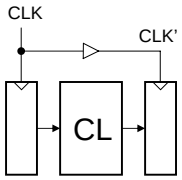
Clock Skew Constraints

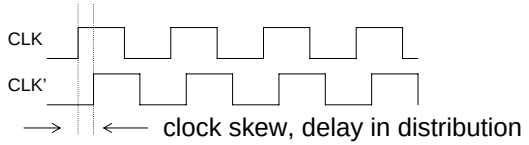




- If clock period T = $T_{CL}+T_{setup}+T_{clk\rightarrow Q}$, circuit will fail
  - Delay relative to CLK = $T_{skew} + T_{CL}+T_{setup}+T_{clk\rightarrow Q}$
- Therefore:
  1. Control clock skew
     a) Careful clock distribution. Equalize path delay from clock source to all clock loads by controlling wires delay and buffer delay.
     b) don't "gate" clocks.
  2. $T \geq T_{CL}+T_{setup}+T_{clk\rightarrow Q}$ + worst case skew.
- Most modern large high-performance chips (microprocessors) control end to end clock skew to a few tenths of a nanosecond.

Hacking Clock Skew





- Note reversed buffer.
- In this case, clock skew actually provides *extra time* (adds to the effective clock period).
- This effect has been used to help run circuits as higher clock rates. Risky business!
  - What happens when reg at end of distribution tree feeds back to earlier reg?

Time to ask clarifying questions

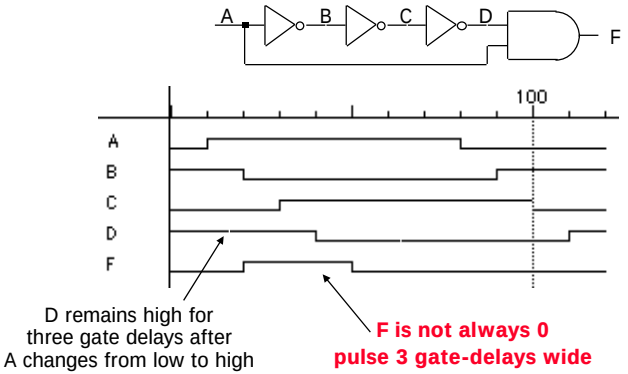
Other effects of Delays on Combinational Logic

Time Behavior of Combinational Networks

- Waveforms
  - Visualization of values carried on signal wires over time
  - Useful in explaining sequences of events (changes in value)
- Simulation tools are used to create these waveforms
  - Input to the simulator includes gates and their connections
  - Input stimulus, that is, input signal waveforms
- Some terms
  - Gate delay—time for change at input to cause change at output
    - » Min delay–typical/nominal delay–max delay
    - » Careful designers design for the worst case
  - Rise time—time for output to transition from low to high voltage
  - Fall time—time for output to transition from high to low voltage
  - Pulse width—time an output stays high or low between changes

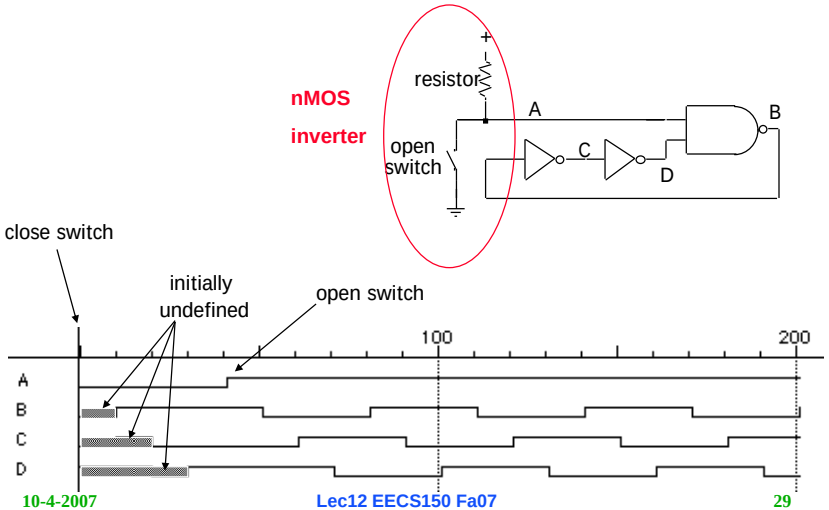
Momentary Changes in Outputs

- Can be useful—pulse shaping circuits
- Can be a problem—incorrect circuit operation (glitches/hazards)
- Example: pulse shaping circuit
  - A' • A = 0
  - delays matter in function



Oscillatory Behavior

- Another pulse shaping circuit



Hazards/Glitches

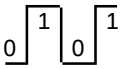
- Hazards/glitches: unwanted switching at the outputs
  - Occur when different paths through circuit have different propagation delays
    - » As in pulse shaping circuits we just analyzed
  - Dangerous if logic causes an action while output is unstable
    - » May need to guarantee absence of glitches
- Usual solutions
  - 1) Wait until signals are stable (by using a clock): preferable (easiest to design when there is a clock – *synchronous* design)
  - 2) Design hazard-free circuits: sometimes necessary (clock not used – *asynchronous* design)

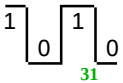
Types of Hazards

- Static 1-hazard
  - Input change causes output to go from 1 to 0 to 1

- Static 0-hazard
  - INput change causes output to go from 0 to 1 to 0

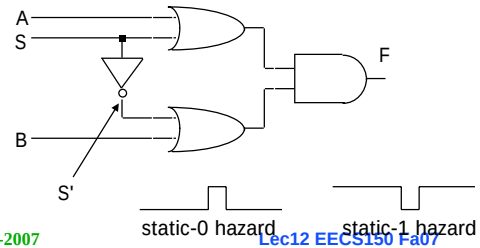
- Dynamic hazards
  - Input change causes a double change from 0 to 1 to 0 to 1 OR from 1 to 0 to 1 to 0

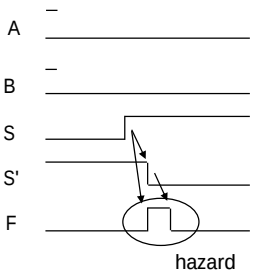




Static Hazards

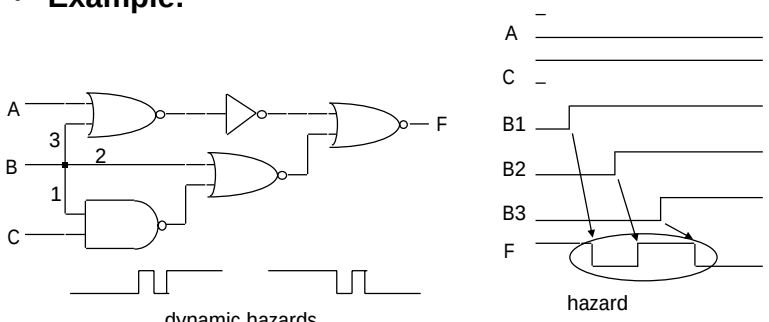
- Due to a literal and its complement momentarily taking on the same value
  - Thru different paths with different delays and converging
- May cause an output that should have stayed at the same value to momentarily take on the wrong value
- Example:



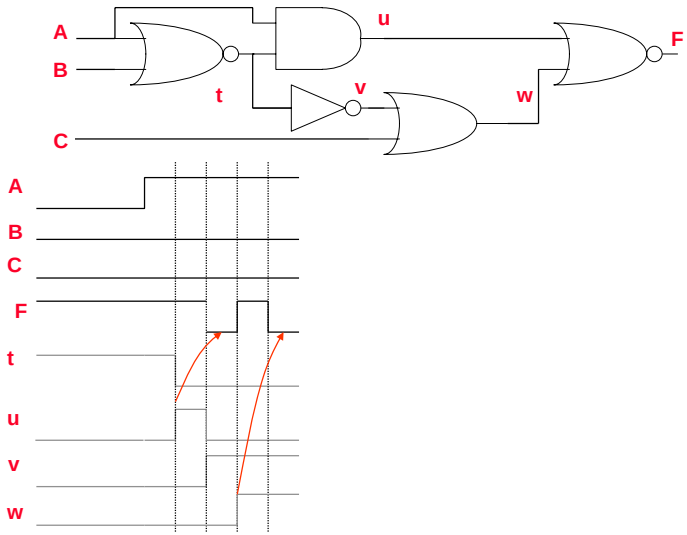


Dynamic Hazards

- Due to the same versions of a literal taking on opposite values
  - Thru different paths with different delays and reconverging
- May cause an output that was to change value to change 3 times instead of once
- Example:

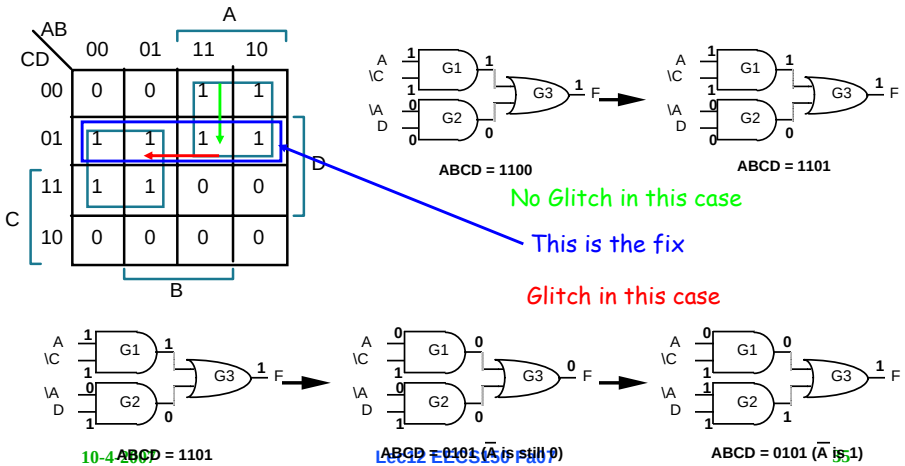


Dynamic Hazards

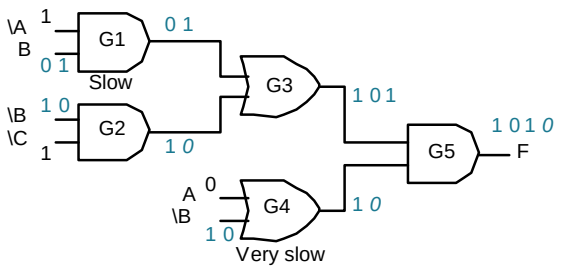


Eliminating Static Hazards

- Following 2-level logic function has a hazard, e.g., when inputs change from ABCD = 0101 to 1101



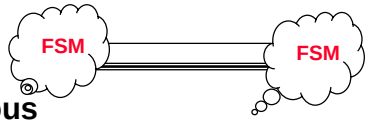
Eliminating Dynamic Hazards



- Very difficult!
- A circuit that is static hazard free can still have dynamic hazards
- Best approach:
  - Design critical circuits to be two level and eliminate all static hazards
  - OR, use good clocked synchronous design style

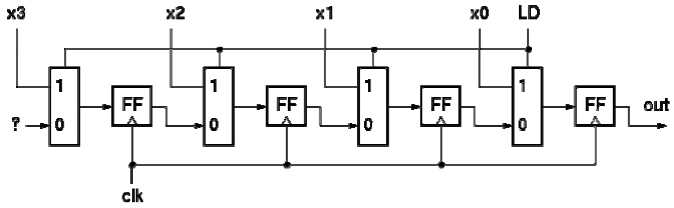
Protocols

- Specified communication and coordination between distinct subsystems
- Realized by cooperating state machines
- Examples everywhere in digital design
  - Rate matching
  - Bus protocols
    - » Memory, chip-to-chip, I/O, …
  - Arbitration for a shared resource
  - Serial protocols
  - Link protocols
  - Network protocols
- Syncronous or asynchronous
- Parallel or serial
- 2-party or multi-party



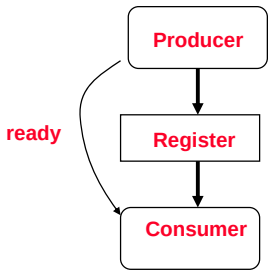
Our old friend…

- Parallel to Serial Converter

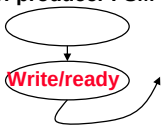


- No "protocol" between FF's
- Every cycle they all move together
- Delays, rates, communication all designed together.

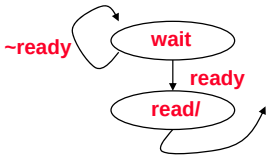
Simple Protocol Example



Fragment of producer FSM

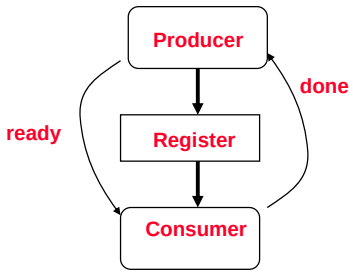


Fragment of consumer FSM

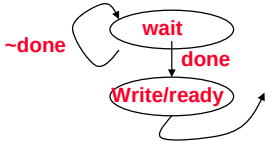


- 1-way communication protocol
- No handshake
- Assumes consumer is always ready to receive

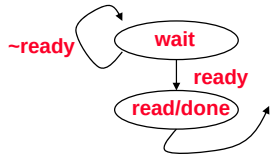
Another Protocol Example



Fragment of producer FSM



Fragment of consumer FSM



- 2-way handshake
- Assumes consumer is always ready to receive

Summary

- All gates have delays
  - RC delay in driving the output
- Wires are distributed RCs
  - Delays goes with the square of the length
- Source circuits determines strength
  - Serial vs parallel
- Delays in combinational logic determine by
  - Input delay
  - Path length
  - Delay of each gate along the path
  - Worst case over all possible input-outputs
- Setup and CLK-Q determined by the two latches in flipflop
- Clock cycle : $T_{cycle} \geq T_{CL}+T_{setup}+T_{clk \rightarrow Q}$ + worst case skew

- Delays can introduce glitches in combinational logic
- Subsystems glued together via protocols
  - Delays, rates, design partitioning