# EECS 150 - Components and Design Techniques for Digital Systems

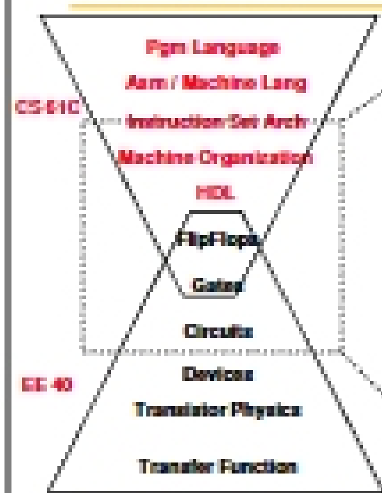
## Lec 27 – Summary (whirlwind)

12-9-04

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

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

## Background




**Deep Digital Design Experience**
- Fundamentals of Boolean Logic
- Synchronous Circuits
- Finite State Machines
- Timing & Clocking
- Device Technology & Implications
- Controller Design
- Arithmetic Units
- Bus Design
- Encoding, Framing
- Testing, Debugging
- Hardware Architecture
- HDL, Design Flow (CAD)



## Course Content

**Components and Design Techniques for Digital Systems**

Synchronous Digital Hardware Systems

- **Synchronous:** "Clocked" - all changes in the system are controlled by a global clock and happen at the same time (not asynchronous)
- **Digital:** All inputs/outputs and internal values (signals) take on discrete values (not analog).

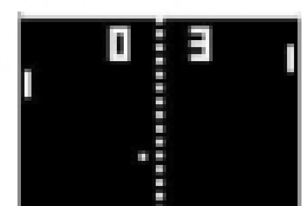
Example digital representation: acoustic waveform

A series of numbers is used to represent the waveform, rather than a voltage or current, as in analog systems.



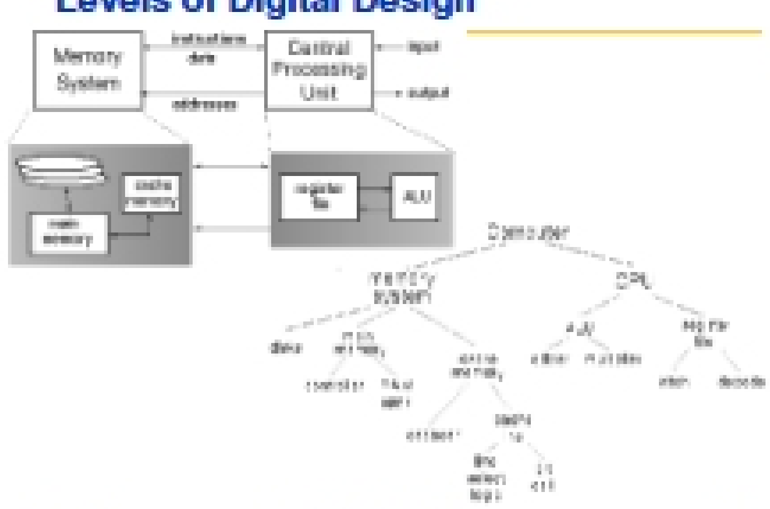
## Trick you into building an extreme project

- FPGA/SDRAM provides full game logic
  - Court, obstructions
  - Moving paddles
  - Moving, colliding ball
  - All the physics
- Court displayed to NTSC (TV) Video Output
  - Real time Sound effects ???
- N64 controller (and switches) for input
- How to make it multiplayer?
  - The network



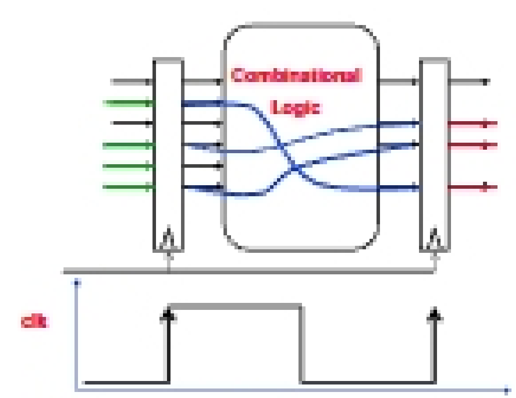


## Levels of Digital Design






## What makes Digital Systems tick?



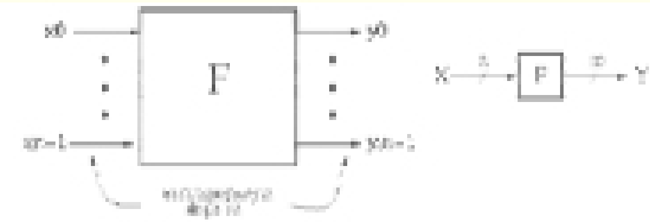

What determines the systems performance?

## The 150 "stuff"

- Building blocks of computer systems
  - ICs (Chips), PCBs, Chassis, Cables & Connectors
- CMOS Transistors
  - Voltage controlled switches
  - Complementary forms (nmos, pmos)
- Logic gates from CMOS transistors
  - Logic gates implement particular boolean functions
    - N inputs, 1 output
  - Serial and parallel switches
  - Dual structure
  - P-type "pull up" transmit 1
  - N-type
- Complex gates: mux
- Synchronous Sequential Elements
  - D FlipFlops



## Combinational Logic (CL) Defined




$y_i = f_i(x0, \ldots, xn\text{-}1)$, where x, y are {0,1}.
Y is a function of only X.

- If we change X, Y will change
  - immediately (well almost!).
  - There is an implementation dependent delay from X to Y.



## Transistor-level Logic Circuits - NAND

- Inverter (NOT gate):
- NAND gate


| a b | out |
|---|---|
| 0 0 | 1 |
| 0 1 | 1 |
| 1 0 | 1 |
| 1 1 | 0 |

nand (out, a, b)

- Logic Function:
  - out = 0 iff both a AND b = 1 therefore out = (ab)'
  - pFET network and nFET network are duals of one another.

How about AND gate?



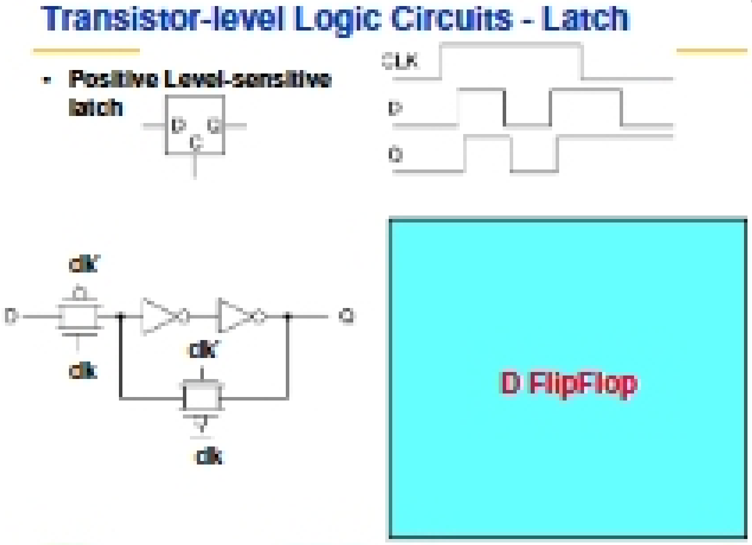
## Combinational logic summary

- Logic functions, truth tables, and switches
  - NOT, AND, OR, NAND, NOR, XOR, . . ., minimal set
- Axioms and theorems of Boolean algebra
  - Proofs by re-writing and perfect induction
- Gate logic
  - Networks of Boolean functions and their time behavior
- Canonical forms
  - Two-level and incompletely specified functions
- Optimization
  - Two-level simplification using K-maps
  - Automation of simplification
  - Multi-level logic
- Later
  - Design case studies
  - Time behavior



## Transistor-level Logic Circuits - Latch

- Positive Level-sensitive latch

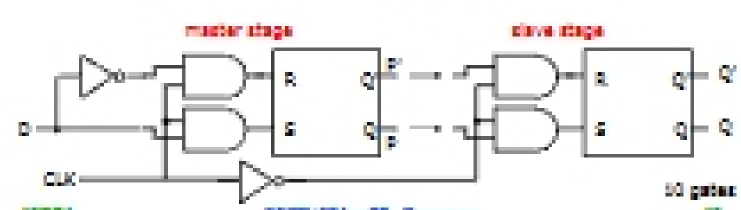



D FlipFlop



## D Flip-Flop

- Make S and R complements of each other in Master stage
  - Eliminates 1s catching problem
    - Input only needs to settle by clock edge
  - Can't just hold previous value (must have new value ready every clock period)
  - Value of D just before clock goes low is what is stored in flip-flop
  - Can make R-S flip-flop by adding logic to make D = S + R' Q



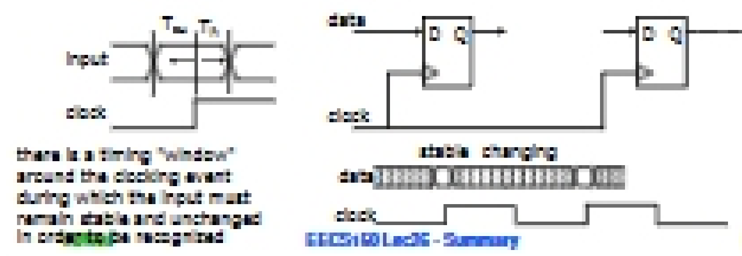
## Timing Methodologies

- Rules for interconnecting components and clocks
  - Guarantee proper operation of system when strictly followed
- Approach depends on building blocks used for memory elements
  - Focus on systems with edge-triggered flip-flops
    - Found in programmable logic devices
  - Many custom integrated circuits focus on level-sensitive latches
- Basic rules for correct timing:
  - (1) Correct inputs, with respect to time, are provided to the flip-flops
  - (2) No flip-flop changes state more than once per clocking event



## Timing Methodologies (cont'd)

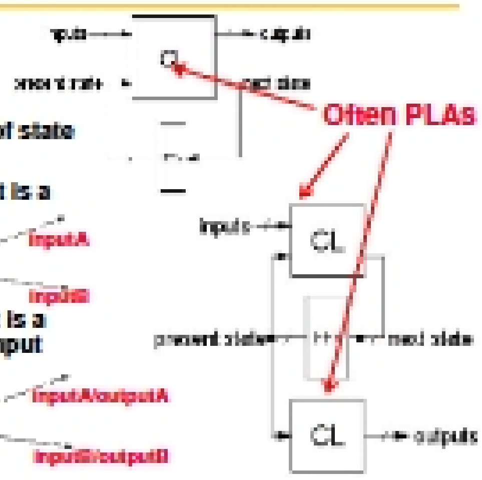
- Definition of terms
  - clock: periodic event, causes state of memory element to change; can be rising or falling edge, or high or low level
  - setup time: minimum time before the clocking event by which the input must be stable (Tsu)
  - hold time: minimum time after the clocking event until which the input must remain stable (Th)



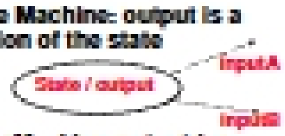

there is a timing "window" around the clocking event during which the input must remain stable and unchanged in order to be recognized



## What's an FSM?




- Next state is function of state and input
- Moore Machine: output is a function of the state




- *Mealy Machine*: output is a function of state and input



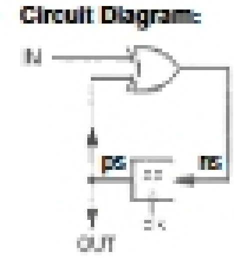

## Formal Design Process for FSMs

Logic equations from table:

OUT = PS

NS = PS xor IN

- Circuit Diagram:

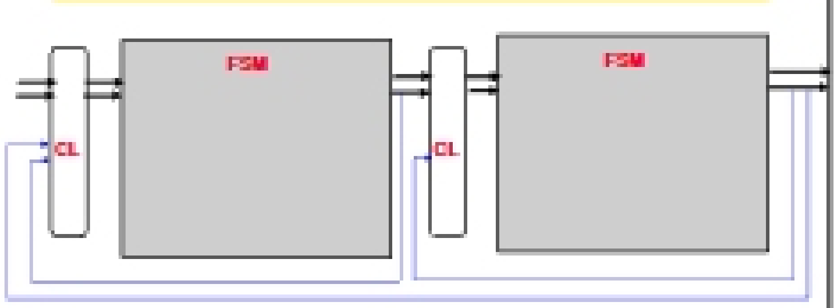



- XOR gate for ns calculation
- DFF to hold present state
- no logic needed for output

- Review of Design Steps:
  1. Circuit functional specification
  2. State Transition Diagram
  3. Symbolic State Transition Table
  4. Encoded State Transition Table
  5. Derive Logic Equations
  6. Circuit Diagram
     - FFs for state
     - CL for NS and OUT



## Composing FSMs into larger designs






## Sequential Synchronous Elements

- Basic registers
  - Common control, MUXes
- Simple, important FSMs
  - simple internal feedback
  - Ring counters, Pattern detectors
  - Binary Counters
- Universal Shift Register
- Using Counters to build controllers
  - Simplify control by controlling simpler FSM