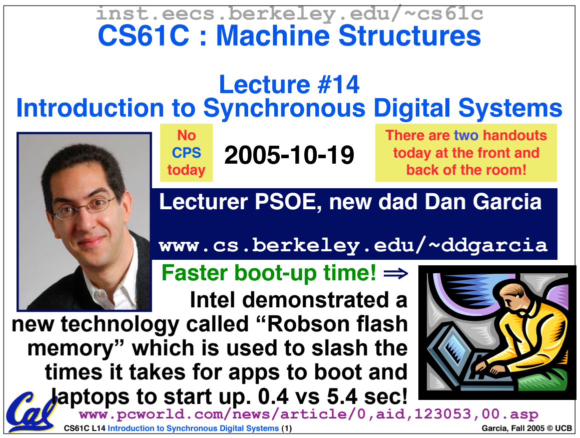


inst.eecs.berkeley.edu/~cs61c

CS61C : Machine Structures

Lecture #14
Introduction to Synchronous Digital Systems

No CPS today

2005-10-19

There are two handouts today at the front and back of the room!

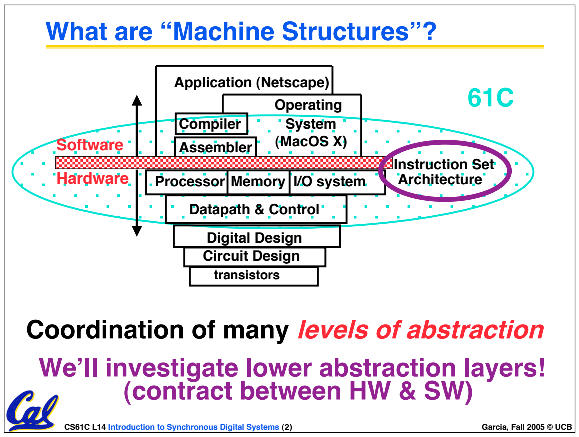
Lecturer PSOE, new dad Dan Garcia

www.cs.berkeley.edu/~ddgarcia

Faster boot-up time! ⇒

Intel demonstrated a new technology called "Robson flash memory" which is used to slash the times it takes for apps to boot and laptops to start up. 0.4 vs 5.4 sec!

www.pcworld.com/news/article/0,aid,123053,00.asp



What are "Machine Structures"?

Application (Netscape)

Operating System (MacOS X)

Compiler

Assembler

Software

Hardware

Instruction Set Architecture

Processor | Memory | I/O system

Datapath & Control

Digital Design

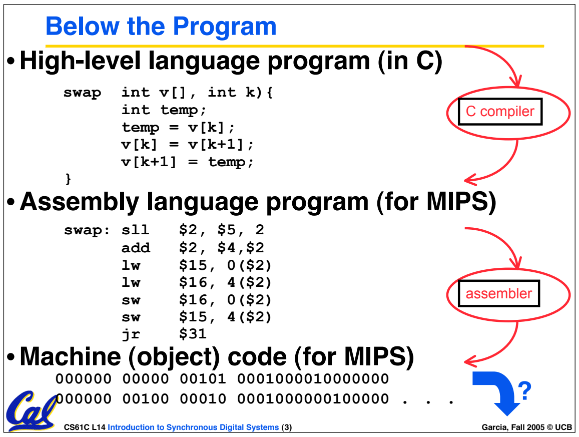
Circuit Design

transistors

61C

Coordination of many *levels of abstraction*

We'll investigate lower abstraction layers! (contract between HW & SW)



Below the Program

- High-level language program (in C)

```
swap  int v[], int k){
      int temp;
      temp = v[k];
      v[k] = v[k+1];
      v[k+1] = temp;
}
```

C compiler

- Assembly language program (for MIPS)

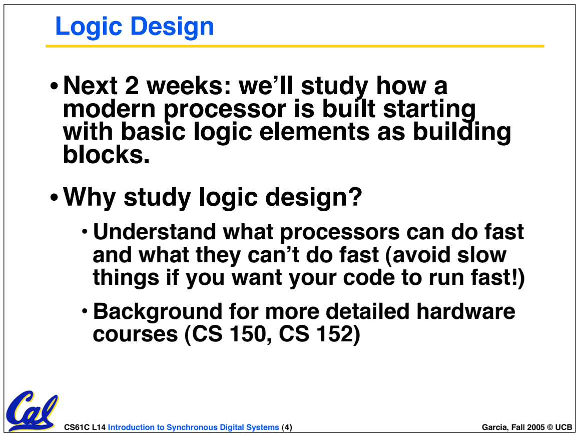
```
swap: sll   $2, $5, 2
      add   $2, $4,$2
      lw    $15, 0($2)
      lw    $16, 4($2)
      sw    $16, 0($2)
      sw    $15, 4($2)
      jr    $31
```

assembler

- Machine (object) code (for MIPS)

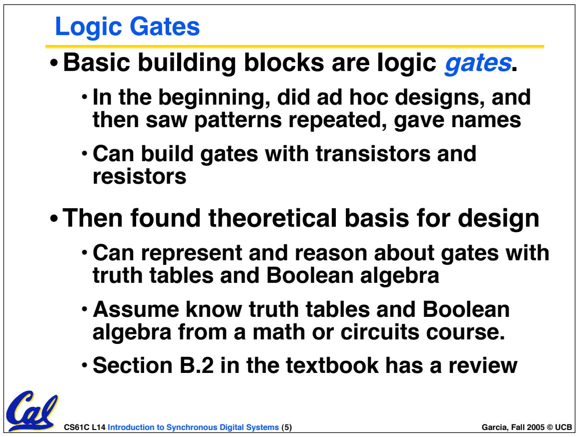
```
000000 00000 00101 0001000010000000
000000 00100 00010 0001000000100000  . . .
```

?



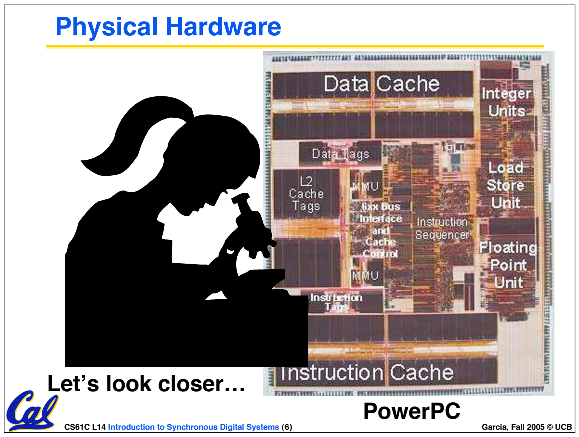
Logic Design

- Next 2 weeks: we'll study how a modern processor is built starting with basic logic elements as building blocks.
- Why study logic design?
  - Understand what processors can do fast and what they can't do fast (avoid slow things if you want your code to run fast!)
  - Background for more detailed hardware courses (CS 150, CS 152)



Logic Gates

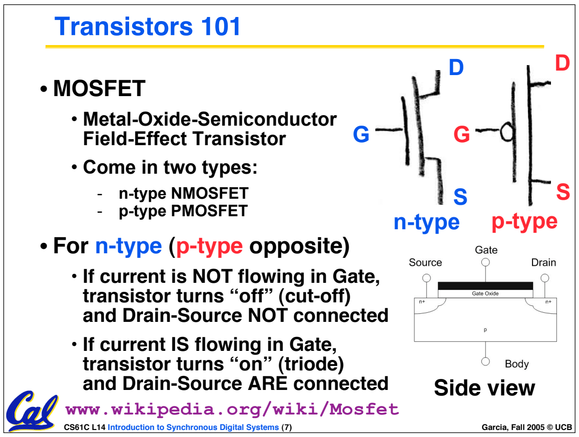
- Basic building blocks are logic *gates*.
  - In the beginning, did ad hoc designs, and then saw patterns repeated, gave names
  - Can build gates with transistors and resistors
- Then found theoretical basis for design
  - Can represent and reason about gates with truth tables and Boolean algebra
  - Assume know truth tables and Boolean algebra from a math or circuits course.
  - Section B.2 in the textbook has a review



Physical Hardware

Let's look closer…

PowerPC

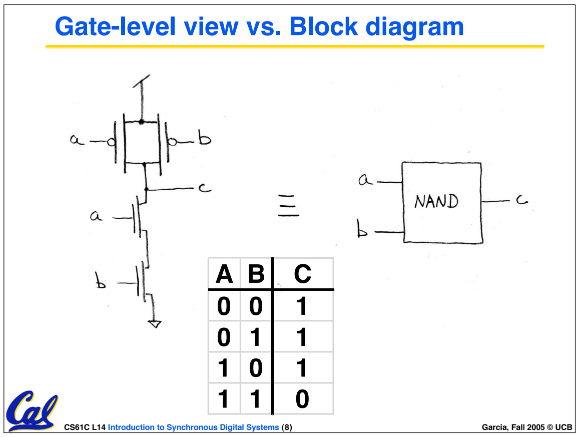


Transistors 101

- MOSFET
  - Metal-Oxide-Semiconductor Field-Effect Transistor
  - Come in two types:
    - n-type NMOSFET
    - p-type PMOSFET
- For n-type (p-type opposite)
  - If current is NOT flowing in Gate, transistor turns "off" (cut-off) and Drain-Source NOT connected
  - If current IS flowing in Gate, transistor turns "on" (triode) and Drain-Source ARE connected

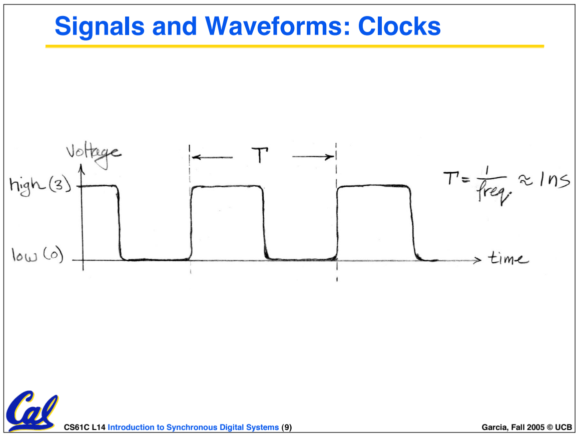
Side view

www.wikipedia.org/wiki/Mosfet

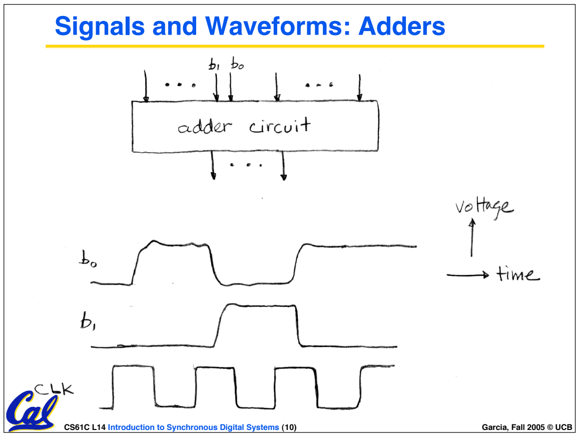


Gate-level view vs. Block diagram

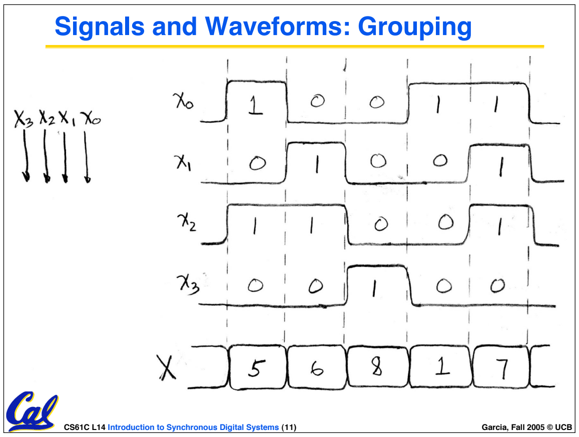
A	B	C
0	0	1
0	1	1
1	0	1
1	1	0



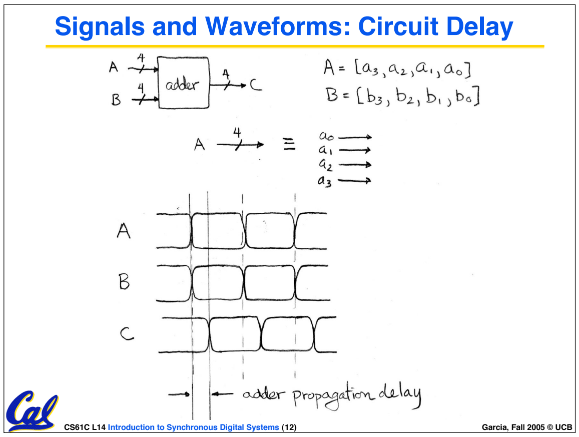
Signals and Waveforms: Clocks



Signals and Waveforms: Adders



Signals and Waveforms: Grouping

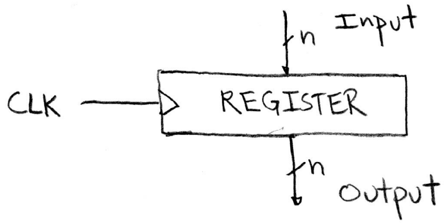


Signals and Waveforms: Circuit Delay

Combinational Logic

- Complex logic blocks are built from basic AND, OR, NOT building blocks we'll see shortly.
- A *combinational* logic block is one in which the output is a function only of its current input.
- Combinational logic cannot have memory (e.g., a register is not a combinational unit).

Circuits with STATE (e.g., register)

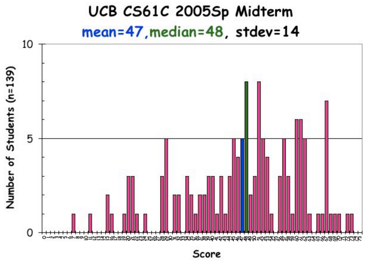


And in semi conclusion…

- ISA is very important abstraction layer
  - Contract between HW and SW
- Basic building blocks are logic *gates*
- Clocks control pulse of our circuits
- Voltages are analog, quantized to 0/1
- Circuit delays are fact of life
- Two types
  - Stateless Combinational Logic (&,|,~)
  - State circuits (e.g., registers)

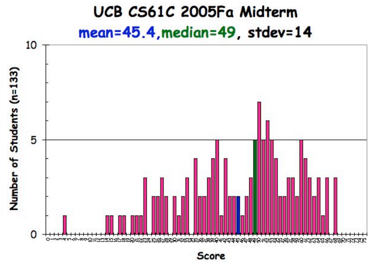
Administrivia - Midterm 2005Sp

- Your TAs and readers stayed up until 4am to get your exams back to you!
- $\bar{x}$: 47, Median: 48, σ: 14.4



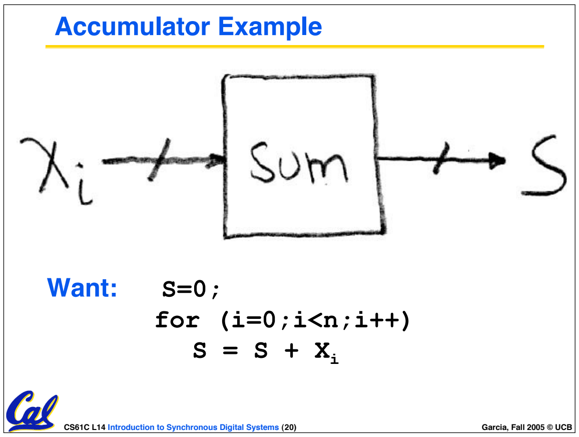
Administrivia - Midterm 2005Fa

- Your TAs and readers stayed up until 4am to get your exams back to you!
- $\bar{x}$: 45.4, Median: 49, σ: 13.7



Administrivia

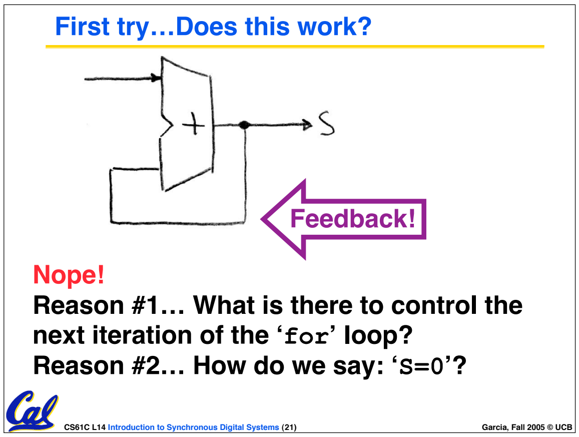
- If you want an exam regrade, simply staple a note to the front of your exam and turn it in to your TA or Dan.
  - We'll collect them until the end of Monday's lecture and then regrade all.
  - Remember that your grade can go down.
- Project 1 is graded; you have one week to request a regrade there too…



Accumulator Example

Want:

```
S=0;
for (i=0;i<n;i++)
    S = S + $X_i$
```

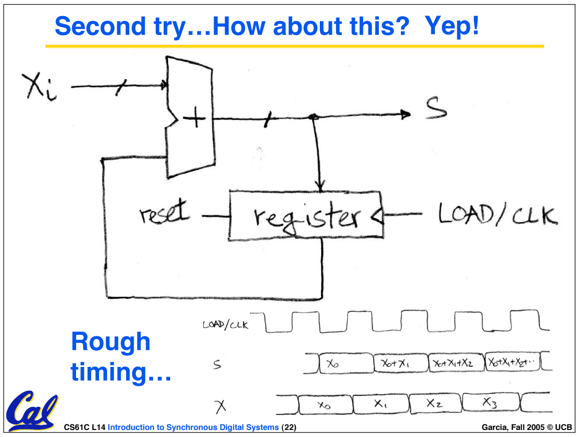


First try…Does this work?

Nope!

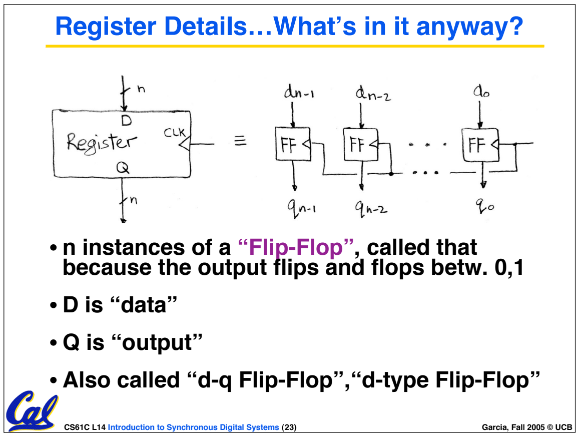
Reason #1… What is there to control the next iteration of the ‘for’ loop?

Reason #2… How do we say: ‘S=0’?



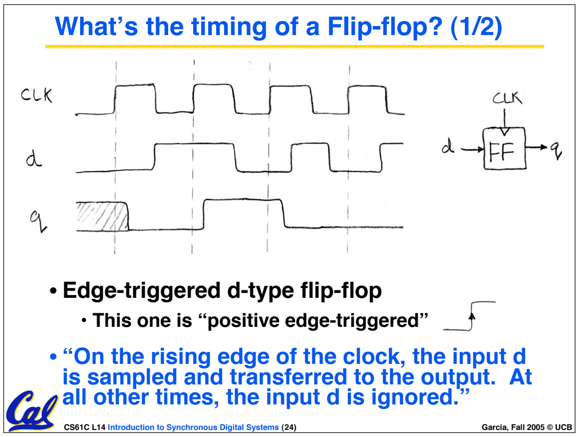
Second try…How about this? Yep!

Rough timing…



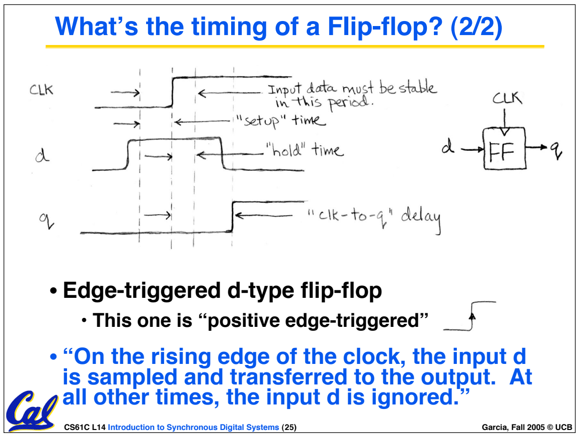
Register Details…What’s in it anyway?

- n instances of a “Flip-Flop”, called that because the output flips and flops betw. 0,1
- D is “data”
- Q is “output”
- Also called “d-q Flip-Flop”,“d-type Flip-Flop”



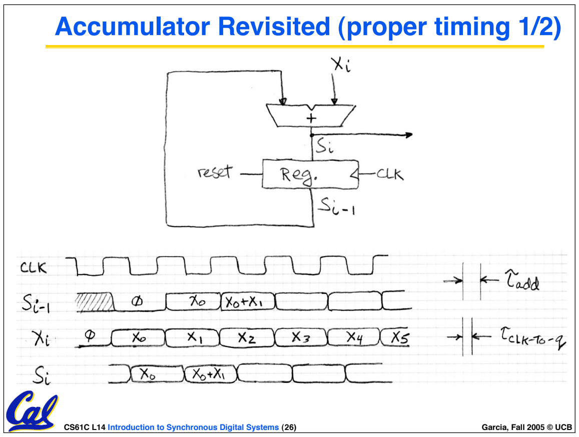
What’s the timing of a Flip-flop? (1/2)

- Edge-triggered d-type flip-flop
  - This one is “positive edge-triggered”
- “On the rising edge of the clock, the input d is sampled and transferred to the output. At all other times, the input d is ignored.”

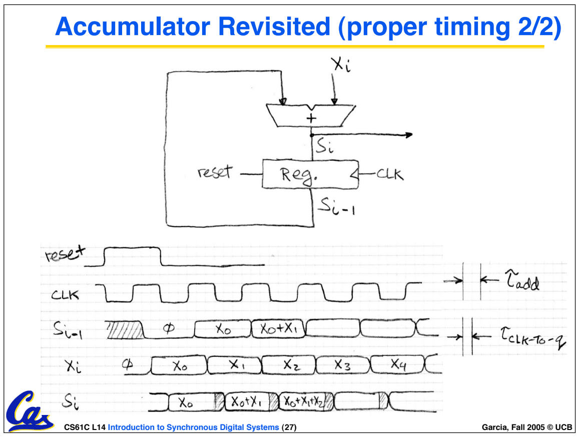


What’s the timing of a Flip-flop? (2/2)

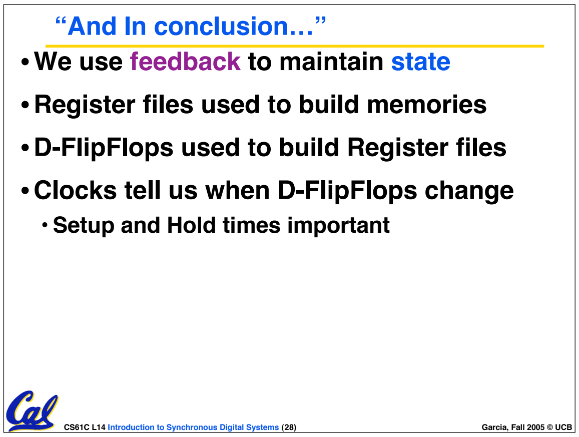
- Edge-triggered d-type flip-flop
  - This one is “positive edge-triggered”
- “On the rising edge of the clock, the input d is sampled and transferred to the output. At all other times, the input d is ignored.”



Accumulator Revisited (proper timing 1/2)



Accumulator Revisited (proper timing 2/2)



“And In conclusion…”

- We use feedback to maintain state
- Register files used to build memories
- D-FlipFlops used to build Register files
- Clocks tell us when D-FlipFlops change
  - Setup and Hold times important