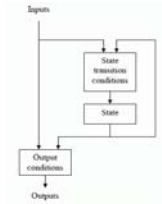


Lecture #14: State and FSMs



2005-07-13

Andy Carle



Outline

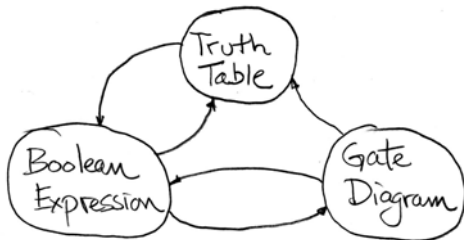
- Waveforms
- State
- Clocks

- FSMs

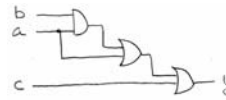


Review (1/3)

- Use this table and techniques we learned to transform from 1 to another



(2/3): Circuit & Algebraic Simplification



original circuit

$$\begin{aligned}
 y &= ((ab) + a) + c \\
 &= ab + a + c \\
 &= a(b + 1) + c \\
 &= a(1) + c \\
 &= a + c
 \end{aligned}$$

equation derived from original circuit

algebraic simplification



simplified circuit



(3/3): Laws of Boolean Algebra

$x \cdot \bar{x} = 0$	$x + \bar{x} = 1$	complementarity
$x \cdot 0 = 0$	$x + 1 = 1$	laws of 0's and 1's
$x \cdot 1 = x$	$x + 0 = x$	identities
$x \cdot x = x$	$x + x = x$	idempotent law
$x \cdot y = y \cdot x$	$x + y = y + x$	commutativity
$(xy)z = x(yz)$	$(x + y) + z = x + (y + z)$	associativity
$x(y + z) = xy + xz$	$x + yz = (x + y)(x + z)$	distribution
$xy + xz = x(y + z)$	$(x + y)x = x$	uniting theorem
$\bar{\bar{x}} = x$	$\overline{(x + y)} = \bar{x} \cdot \bar{y}$	DeMorgan's Law



Signals and Waveforms

- Outputs of CL change over time
 - With what? → Change in inputs

- Can graph changes with waveforms ...



Signals and Waveforms: Adders

CS 61C L14 State (7)

Signals and Waveforms: Grouping

CS 61C L14 State (8)

Signals and Waveforms: Circuit Delay

CS 61C L14 State (9)

State

- With CL, output is always a function of **CURRENT** input
 - With some (variable) propagation delay
- Clearly, we need a way to introduce state into computation

CS 61C L14 State (10)

Accumulator Example

Want: $S=0$; for i from 0 to $n-1$
 $S = S + X_i$

CS 61C L14 State (11)

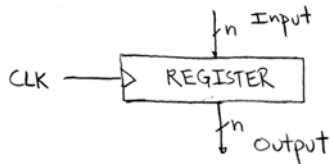
First try...Does this work?

Nope!
 Reason #1... What is there to control the next iteration of the 'for' loop?
 Reason #2... How do we say: 's=0'?

Need a way to store partial sums! ...

CS 61C L14 State (12)

Circuits with STATE (e.g., register)



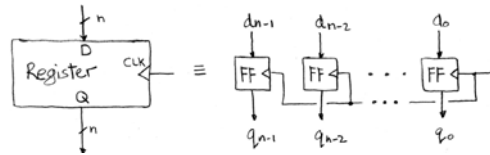
- Need a Logic Block that will:
1. store output (partial sum) for a while,
 2. until we tell it to update with a new value.

Cal

CS 61C L14 State (13)

A. Carls, Summer 2005 © UCB

Register Details...What's in it anyway?



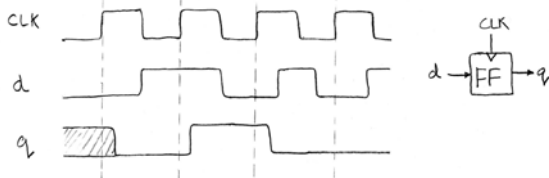
- n instances of a "Flip-Flop", called that because the output flips and flops betw. 0,1
- D is "data"
- Q is "output"
- Also called "d-q Flip-Flop", "d-type Flip-Flop"

Cal

CS 61C L14 State (14)

A. Carls, Summer 2005 © UCB

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



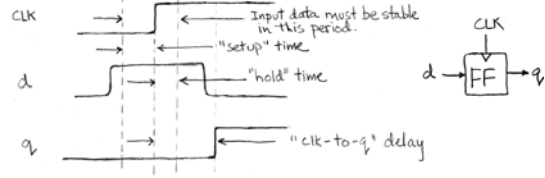
- Edge-triggered D-type flip-flop
- This one is "positive edge-triggered"
- "On the rising edge of the clock, the input d is sampled and transferred to the output. At all other times, the input d is ignored."

Cal

CS 61C L14 State (15)

A. Carls, Summer 2005 © UCB

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



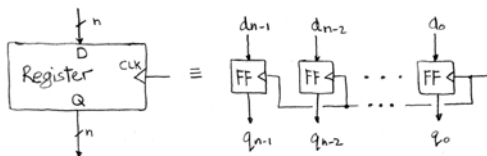
- Edge-triggered D-type flip-flop
- This one is "positive edge-triggered"
- "On the rising edge of the clock, the input d is sampled and transferred to the output. At all other times, the input d is ignored."

Cal

CS 61C L14 State (16)

A. Carls, Summer 2005 © UCB

Bus a bunch of D FFs together ...



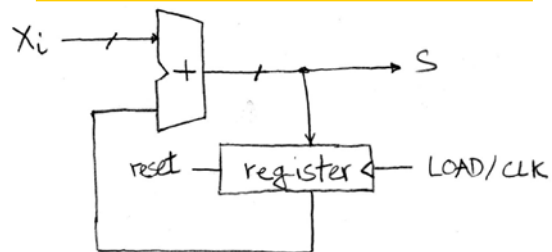
- Register of size N:
- n instances of D Flip-Flop

Cal

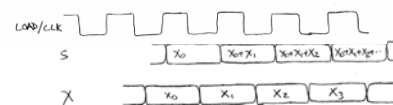
CS 61C L14 State (17)

A. Carls, Summer 2005 © UCB

Second try...How about this? Yep!



Rough timing...

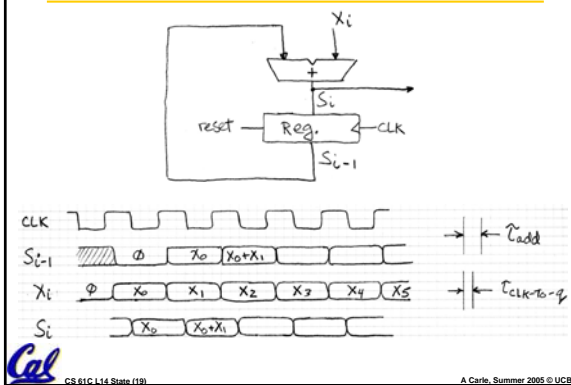


Cal

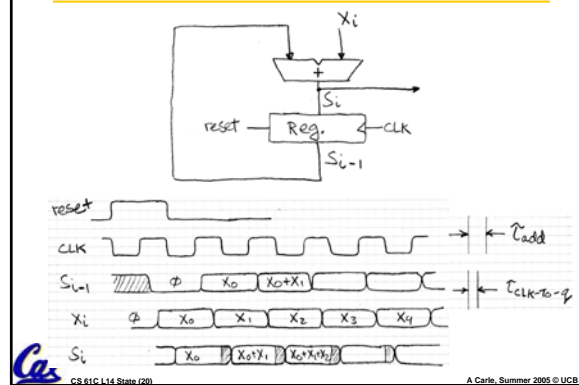
CS 61C L14 State (18)

A. Carls, Summer 2005 © UCB

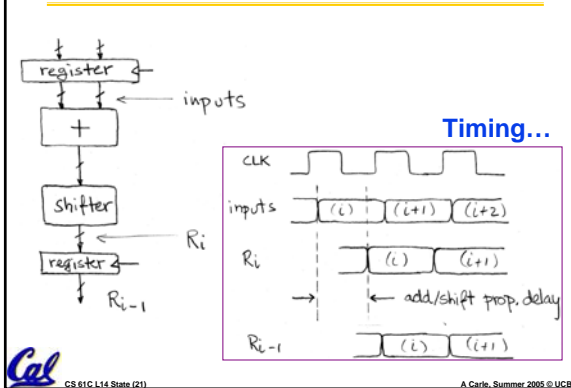
Accumulator Revisited (proper timing 1/2)



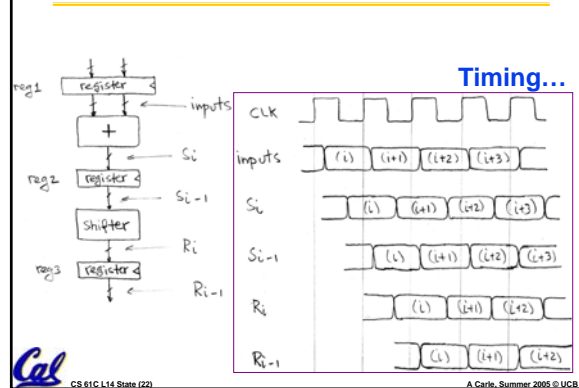
Accumulator Revisited (proper timing 2/2)



Pipelining to improve performance (1/2)



Pipelining to improve performance (2/2)



Peer Instruction 1

• Simplify the following Boolean algebra equation:

$$Q = !(A * B) + !(A * C)$$

• Use algebra, individual steps, etc.

• Don't just look at it and figure it out, or I'll have to start using harder examples. ☹

Administrivia

• HW 45 due Monday


• Project 2 will be released soon

• If you want to get a little bit ahead (in a moderately fun sort of way), start playing with Logisim:

• <http://ozark.hendrix.edu/~burch/logisim/>

Clocks

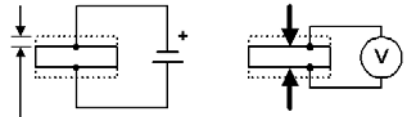
- Need a regular oscillator:



- Wire up three inverters in feedback?...
 - Not stable enough
 - 1->0 and 0->1 transitions not symmetric.
- Solution: Base oscillation on a natural resonance. But of what?

Cal CS 61C L14 Slide (26) A Carls, Summer 2005 © UCB

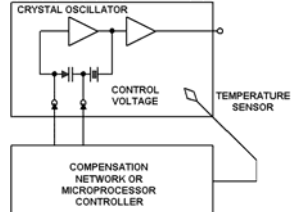
Clocks



- Crystals and the Piezoelectric effect:
 - Voltage \rightarrow deformation \rightarrow voltage \rightarrow ...
 - Deformations have a resonant freq.
 - Function of crystal cut

Cal CS 61C L14 Slide (28) A Carls, Summer 2005 © UCB

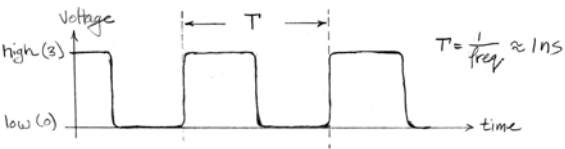
Clocks



- Controller puts AC across crystal:
 - At anything but resonant freqs \rightarrow destructive interference
 - Resonant freq \rightarrow CONSTRUCTIVE!

Cal CS 61C L14 Slide (27) A Carls, Summer 2005 © UCB

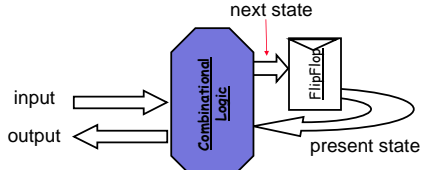
Signals and Waveforms: Clocks



Cal CS 61C L14 Slide (29) A Carls, Summer 2005 © UCB

FSMs

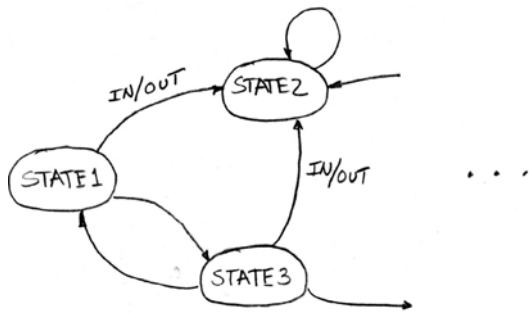
- With state elements, we can build circuits whose output is a function of inputs and current state.



- State transitions will occur on clock edges.

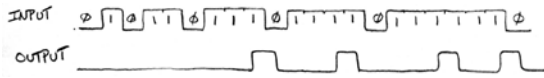
Cal CS 61C L14 Slide (29) A Carls, Summer 2005 © UCB

Finite State Machines Introduction

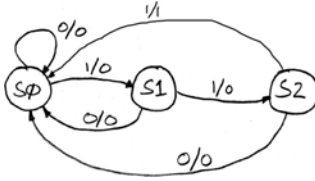


Cal CS 61C L14 Slide (30) A Carls, Summer 2005 © UCB

Finite State Machine Example: 3 ones...



Draw the FSM...



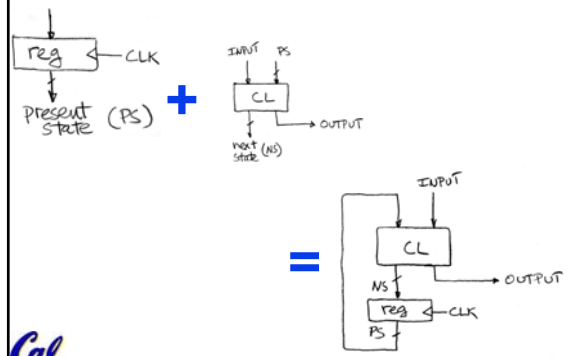
PS	Input	NS	Output
00	0	00	0
00	1	01	0
01	0	00	0
01	1	10	0
10	0	00	0
10	1	00	1

Cal

CS 61C L14 State (31)

A. Carls, Summer 2005 © UC Berkeley

Hardware Implementation of FSM

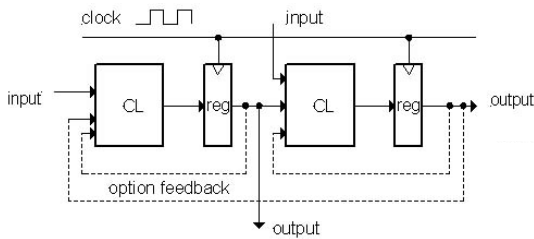


Cal

CS 61C L14 State (32)

A. Carls, Summer 2005 © UC Berkeley

General Model for Synchronous Systems



Cal

CS 61C L14 State (33)

A. Carls, Summer 2005 © UC Berkeley

Peer Instruction 2

- Two bit counter:
 - 4 States: 0, 1, 2, 3
 - When input c is high, go to next state - (3->0)
 - When input is low, don't change state
 - On the transition from state 3 to state 0, output a 1. At all other times, output 0.

Cal

CS 61C L14 State (34)

A. Carls, Summer 2005 © UC Berkeley