

inst.eecs.berkeley.edu/~cs61c
UC Berkeley CS61C : Machine Structures

Lecture 21 – State Elements :
 Circuits that Remember

2007-03-07

“Mocha sipping” TA Valerie Ishida

inst.eecs.berkeley.edu/~cs61c-td

161 Exabytes In 2006 ⇒

In 2006 we created, captured, and replicated 161 exabytes of digital information. The last time anyone tried to estimate global information volume, in 2003, researchers at UC Berkeley came up with 5 exabytes. By 2010 almost a zettabyte!



<http://hardware.slashdot.org/article.pl?id=07/03/05/2355246>

CS61C L21 State Elements - Circuits that Remember (1) Spring 2007 © UCB

Review

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

CS61C L21 State Elements - Circuits that Remember (2) Spring 2007 © UCB


Uses for State Elements

1. As a place to store values for some indeterminate amount of time:
 - Register files (like \$1-\$31 on the MIPS)
 - Memory (caches, and main memory)
2. Help control the flow of information between combinational logic blocks.
 - State elements are used to hold up the movement of information at the inputs to combinational logic blocks and allow for orderly passage.

CS61C L21 State Elements - Circuits that Remember (3) Spring 2007 © UCB

Accumulator Example

Why do we need to control the flow of information?



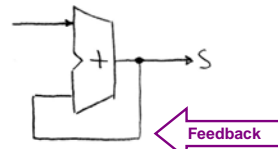
Want: $s=0;$
 for ($i=0; i<n; i++$)
 $S = S + X_i$

Assume:

- Each X value is applied in succession, one per cycle.
- After n cycles the sum is present on S.

CS61C L21 State Elements - Circuits that Remember (4) Spring 2007 © UCB

First try...Does this work?



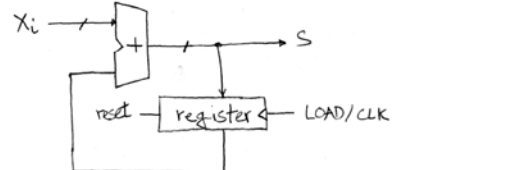
Nope!

Reason #1... What is there to control the next iteration of the 'for' loop?

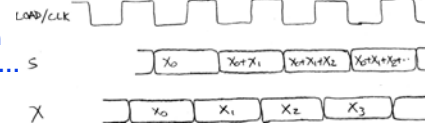
Reason #2... How do we say: 's=0'?

CS61C L21 State Elements - Circuits that Remember (5) Spring 2007 © UCB

Second try...How about this?



Rough timing...



Register is used to hold up the transfer of data to adder.

CS61C L21 State Elements - Circuits that Remember (6) Spring 2007 © UCB

Register Details...What's inside?

- n instances of a "Flip-Flop"
- Flip-flop name because the output flips and flops between 0,1
- D is "data", Q is "output"
- Also called "d-type Flip-Flop"

Cal CS61C L21 State Elements - Circuits that Remember (7) Spring 2007 © 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."
- Example waveforms:

Cal CS61C L21 State Elements - Circuits that Remember (8) Spring 2007 © 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."
- Example waveforms (more detail):

Cal CS61C L21 State Elements - Circuits that Remember (9) Spring 2007 © UCB

Administrivia -

- **Midterms graded!!**
 - handed back in Dis/Lab
 - TAs and TAsIT graded from 10 PM to 6 AM!
 - DO NOT PANIC (remember clobbering)
 - You can get a 0 on this midterm and still get the highest overall grade in the class!

Cal CS61C L21 State Elements - Circuits that Remember (10) Spring 2007 © UCB

Accumulator Revisited (proper timing 1/2)

- Reset input to register is used to force it to all zeros (takes priority over D input).
- S_i holds the result of the i^{th} -1 iteration.
- Analyze circuit timing starting at the output of the register.

Cal CS61C L21 State Elements - Circuits that Remember (11) Spring 2007 © UCB

Accumulator Revisited (proper timing 2/2)

- reset signal shown.
- Also, in practice X might not arrive at the adder at the same time as S_{i-1}
- S_i temporarily is wrong, but register always captures correct value.
- In good circuits, instability never happens around rising edge of clk.

Cal CS61C L21 State Elements - Circuits that Remember (12) Spring 2007 © UCB

Pipelining to improve performance (1/2)

Extra Register are often added to help speed up the clock rate.

Timing...

Note: delay of 1 clock cycle from input to output.
Clock period limited by propagation delay of adder/shifter.

CS61C L21 State Elements - Circuits that Remember (13) Spring 2007 © UC Berkeley

Pipelining to improve performance (2/2)

- Insertion of register allows higher clock frequency.
- More outputs per second.

Timing...

CS61C L21 State Elements - Circuits that Remember (14) Spring 2007 © UC Berkeley

Finite State Machines (FSM) Introduction

- You have seen FSMs in other classes.
- Same basic idea.
- The function can be represented with a "state transition diagram".
- With combinational logic and registers, any FSM can be implemented in hardware.

CS61C L21 State Elements - Circuits that Remember (15) Spring 2007 © UC Berkeley

Finite State Machine Example: 3 ones...

FSM to detect the occurrence of 3 consecutive 1's in the input.

Draw the FSM...

Assume state transitions are controlled by the clock:
on each clock cycle the machine checks the inputs and moves to a new state and produces a new output...

CS61C L21 State Elements - Circuits that Remember (16) Spring 2007 © UC Berkeley

Hardware Implementation of FSM

... Therefore a register is needed to hold the a representation of which state the machine is in. Use a unique bit pattern for each state.

Combinational logic circuit is used to implement a function maps from present state and input to next state and output.

CS61C L21 State Elements - Circuits that Remember (17) Spring 2007 © UC Berkeley

Hardware for FSM: Combinational Logic

Next lecture we will discuss the detailed implementation, but for now can look at its functional specification, truth table form.

Truth table...

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

CS61C L21 State Elements - Circuits that Remember (18) Spring 2007 © UC Berkeley

General Model for Synchronous Systems

- Collection of CL blocks separated by registers.
- Registers may be back-to-back and CL blocks may be back-to-back.
- Feedback is optional.
- Clock signal(s) connects only to clock input of registers.

CS61C L21 State Elements - Circuits that Remember (19) Spring 2007 © UCB

Peer Instruction

A. HW feedback akin to SW recursion

B. We can implement a D-Q flipflop as simple CL (And, Or, Not gates)

C. You can build a FSM to signal when an equal number of 0s and 1s has appeared in the input.

ABC
1: FFF
2: FFT
3: FTF
4: FTT
5: TFF
6: TFT
7: TTF
8: TTT

CS61C L21 State Elements - Circuits that Remember (20) Spring 2007 © UCB

Peer Instruction Answer

A. It needs 'base case' (reg reset), way to step from i to $i+1$ (use register + clock).

- True!

B. D-Q has state, CL never has state!

- False!

C. How many states would it have? Say it's n . How does it know when $n+1$ bits have been seen?

- False!

A. HW feedback akin to SW recursion

B. We can implement a D-Q flipflop as simple CL (And, Or, Not gates)

C. You can build a FSM to signal when an equal number of 0s and 1s has appeared in the input.

ABC
1: FFF
2: FFT
3: FTF
4: FTT
5: TFF
6: TFT
7: TTF
8: TTT

CS61C L21 State Elements - Circuits that Remember (21) Spring 2007 © UCB

"And In conclusion..."

- State elements are used to:
 - Build memories
 - Control the flow of information between other state elements and combinational logic
- D-flip-flops used to build registers
- Clocks tell us when D-flip-flops change
 - Setup and Hold times important
- We pipeline long-delay CL for faster clock
- Finite State Machines extremely useful

You'll see them again (150,152) & 164

CS61C L21 State Elements - Circuits that Remember (22) Spring 2007 © UCB