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

\[ y = ((ab) + a) + c \]
\[ = ab + a + c \]
\[ = a(b + 1) + c \]
\[ = a(1) + c \]
\[ = a + c \]
\[ \alpha \]

simplified circuit

(3/3): Laws of Boolean Algebra

Signals and Waveforms

• Outputs of CL change over time
  • With what? \( \rightarrow \) Change in inputs

  • Can graph changes with waveforms …
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

Accumulator Example

Want: \( S = 0 \); for \( i \) from 0 to \( n-1 \)  
\[ S = S + X_i \]

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: ‘\( S = 0 \)’?  
Need a way to store partial sums! ...
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.

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

- n instances of a “Flip-Flop”, called that
because the output flips and flops between 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.”

Bus a bunch of D FFs together...

- Register of size \( N \):
  - \( n \) instances of D Flip-Flop

Second try...How about this? Yep!

- Rough timing...
Peer Instruction 1

• Simplify the following Boolean algebra equation:

  \[ Q = \neg(A \cdot B) + \neg(A \cdot C) \]

  • 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?

Clocks

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

Clocks

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

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.
Finite State Machine Example: 3 ones...

Draw the FSM...

<table>
<thead>
<tr>
<th>PS</th>
<th>Input</th>
<th>NS</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0</td>
<td>00</td>
<td>0</td>
</tr>
<tr>
<td>00</td>
<td>1</td>
<td>01</td>
<td>0</td>
</tr>
<tr>
<td>01</td>
<td>0</td>
<td>00</td>
<td>0</td>
</tr>
<tr>
<td>01</td>
<td>1</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>00</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>00</td>
<td>1</td>
</tr>
</tbody>
</table>

Hardware Implementation of FSM

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.