CS 2630
Computer Organization
Sequential logic
Brandon Myers
University of Iowa
Ingredients for a processor

• Combinational logic
  • arithmetic and bitwise operations
  • multiplexors (pick 1 out of N inputs)
  • control logic
  • branch calculation
  • address calculation

• Synchronous logic
  • registers
  • program counter, or PC
  • data memory
  • instruction memory
Basic memory element: flip-flop

- on the rising edge of the clock signal, the value on input D is "captured" into the flip-flop

- value of the output Q is the value stored in the flip-flop

see this week’s readings for how to implement a flip-flop out of logic gates
Clock signal

- Analogy with a clock on the wall:
  - every *tick* happens “instantaneously” at the *rising edge*
  - the *period* is the time between ticks
- period of a wall clock: 1 second
- period of a typical digital logic clock: nanoseconds or less
Peer instruction

What is the period (in seconds) of this clock signal?

What is the frequency (in Hz) of this clock signal?
Reading waveforms

- at the rising edge, sample D’s current value
- some small delay later, Q becomes that value
Peer instruction

Draw the rest of signal Q. The initial line means that the Flip-flop starts with the value 1.
Administrivia

- Debug Your Brain today instead of Tuesday, this week only
- guest instructor Tu-Th: Dhruv Vyas
- Tuesday: lab 5 on sequential logic
  - (pre lab due tomorrow before class starts)
- Wednesday: Team packet on memories and building your first MIPS instruction
- Thursday: Introduction to MIPS processor implementation

- Project 2 assigned this week (will be 2 parts)
  - start thinking of your project partners (teams of 3)
Register: collection of N flip-flops to store N bits

- Example: 2-bit register

The notation means the number of bits on that wire.
Building an accumulator

i.e., a circuit that would add a sequence of numbers, like this code

```java
int[] A = {5, 1, 3, 1};
int sum = 0;
for (int t=0; t<N; t++) {
    sum += A_t;
}
```

first attempt:

- how do we know when to change $A_t$ to $A_{t+1}$?
- what if some bits of sum change faster than others?
- how do we initialize sum to 0?
Building an accumulator

i.e., a circuit that would add a sequence of numbers, like this code

```c
int[] A = {5, 1, 3, 1};
int sum = 0;
for (int t=0; t<N; t++) {
    sum += A_t;
}
```

second attempt:

![Diagram of an accumulator circuit]
Peer instruction

X_t → 32-bit + → D Reg → Out

<table>
<thead>
<tr>
<th>clk</th>
<th>X</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>?</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>
Peer instruction

Implement a sequential logic circuit that outputs the Fibonacci numbers* 1,1,2,3,5,8,... . The output should be 32 bits and should change once per clock period, like an accumulator. You may use 32-bit adders and registers and you may assume the registers can start holding any value you want.

(work this problem in a group of 2-3 at the board)

Test your circuit at the board (should be a different team member than who drew the circuit).

* next number is sum of the previous two
Difference between combinational and sequential logic?

One way to answer: for each logic, describe the outputs in terms of the inputs
Propagation delay in combinational logic: shown with waveforms

A  B  
\[ \rightarrow \]  
\[ + \rightarrow \text{cout} \]

Sum  
\[ \rightarrow \]  
\[ \text{Sum and cout} \text{ stable here} \]

new A=1 arrives

new B=1 arrives
Summary

• A flip-flop stores 1 bit
• The flip-flop only takes on a new value at the *rising edge of the clock*
• An N-bit register can be built out of N flip-flops
• An accumulator uses feedback to increment the current stored value
  • at some level, the MIPS processor is a more sophisticated, programmable accumulator