Future of Human ⇒ transportation? This article speaks to it. Ultralights may someday rule the air. You pull the wings off & drive away!

http://news.bbc.co.uk/1/hi/magazine/3580337.stm
Review…

- 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 (&, |, ~), in which output is function of input only
  - State circuits (e.g., registers)
Accumulator Example

Want: $S = 0; \text{ for } i \text{ from } 0 \text{ to } n-1$

$S = S + X_i$
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’?
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.”
Administrivia

- Midterm regrades will be done at tonight’s TA mtg and online tomorrow.

- Anyone who cannot make the final exam (religious or graduation reasons only) must email me <ddgarcia@cs> with the subject “61C FINAL EXAM” before the start of next week (Monday).
Accumulator Revisited (proper timing 1/2)

CLK

Si-1

Xi

Si

reset

Reg.

CLK

X_i

S_i

S_i-1

X_0

X_1

X_2

X_3

X_4

X_5

X_0 + X_1

T_{add}

T_{CLK-to-q}
Accumulator Revisited (proper timing 2/2)
Pipelining to improve performance (1/2)

Timing…
Pipelining to improve performance (2/2)

Timing…
Finite State Machines Introduction
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>
“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
• We pipeline big-delay CL for faster clock
• Finite State Machines extremely useful
  • You’ll see them in HW classes (150,152) & 164