Lecture 6
Sequential Circuits and Memory

Xuan ‘Silvia’ Zhang
Washington University in St. Louis

http://classes.engineering.wustl.edu/ese566/
Outline

Combinational Logic

Interconnect

Sequential State
SR Latch

- Basic NOR latch

\[ Q \]

\[ \overline{Q} \]

\[ S \] (set)

\[ R \] (reset)

\[ 0 \]

\[ 1 \]

\[ t_{pd} \]

No change

not allowed

unstable
Other SR Latches

- Clocked

- NAND SR latch
D Latch

• Truth table

SR latch:

<table>
<thead>
<tr>
<th>S</th>
<th>R</th>
<th>Q⁺</th>
<th>Q⁺</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>hold,</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

D latch

<table>
<thead>
<tr>
<th>D</th>
<th>Q(t+1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Level-Sensitive Latch
Flip-Flop

- **Latch timing issue**
  - transparent when $C = 1$
  - state should change only once every new clock cycle

- **Master-slave flip flop**
  - break feedthrough
Flip-Flop Timing Issue

- 1' catching

C

S

R

Y

Master out

Q

Slave out

Master active

Slave active

Glitch

1' catching

wrong output should have been 0
Edge-Triggered D Flip-Flop (DFF)

- Why edge trigger?
- D replace S and R input
Edge-Sensitive Flip-Flop
Outline

Sequential Circuits

Timing Analysis

Memory
Timing in Digital Logic

- Setup time
- Hold time
Timing in Digital Logic

- Launch edge and latch edge
Timing in Digital Logic

- Data arrival time: using launch edge

![Timing Diagram]

Data Arrival Time = launch edge + $T_{clk1} + T_{co} + T_{data}$

<table>
<thead>
<tr>
<th>Tclk</th>
<th>Clock Skew</th>
<th>Tco</th>
<th>FF Clock- &gt; Output</th>
<th>Tdata</th>
<th>Logic Delay</th>
</tr>
</thead>
</table>
Timing in Digital Logic

- Clock arrival time

Clock Arrival Time = latch edge + $T_{clk2}$
Timing in Digital Logic

- Data required time (setup): latch edge

Data Required Time = Clock Arrival Time - $T_{su}$ - Setup Uncertainty
Timing in Digital Logic

- Data required time (hold): next launch = latch

Data Required Time = Clock Arrival Time + $T_h$ + Hold Uncertainty
Timing in Digital Logic

• Setup slack

Setup Slack = Data Required Time – Data Arrival Time

Positive slack
Timing requirement met

Negative slack
Timing requirement not met
Timing in Digital Logic

- Hold slack

Hold Slack = Data Arrival Time – Data Required Time

Positive slack
Timing requirement met

Negative slack
Timing requirement not met
Static Timing Analysis

- Timing Model and Timing Constraint
- Arrival Time (AT) and Required Time (RT)
Outline

Sequential Circuits

Timing Analysis

Memory
Static RAM

- **Applications**
  - CPU register file, cache, embedded memory, DSP

- **Characteristics**
  - 6 transistor per cell, other topologies
  - no need to refresh
  - access time ~ cycle time
  - no charge to leak
  - faster, more area, more expensive
SRAM Operation

• Standby
  - word line de-asserted

• Read
  - precharge bit lines
  - assert WL
  - BL rise/drop slightly

• Write
  - apply value to BL
  - assert WL
  - input drivers stronger
SRAM Architecture

Column Drivers

n-bit word

Bitcell Array

Address Decoder

Bitlines for bit n

Sense Amplifiers

source: semiengineering.com
Multi-Bank Layout

source: semiengineering.com
Questions?

Comments?

Discussion?
Design Tool Tutorials

- Standard-cell based design flow
Design Tool Tutorials

- **Functional Simulation**
  - tool: Synopsys VCS
  - simulate your HDL (eg. Verilog) code to verify functionality

- **Logic Synthesis**
  - tool: Synopsys Design Compiler (DC)
  - convert/synthesize behavioral/RTL level HDL to gate-level netlist (i.e. connectivity list)

- **Physical Design (Place & Route)**
  - tool: Cadence Encounter
  - given the gate-level netlist, place and route the design to complete an IC chip in its final physical form
Acknowledgement

Cornell University, ECE 5745