Sequential Logic:
Clocks
Latches
Flip-Flops
Announcements

• Lab 3 will be released tonight
  – Including a tutorial on Quartus/Verilog

• Late submission policy
  – Email ece2300-staff@csl.cornell.edu before the deadline to request slip days
Example: Logic Function using Mux

- Use a 4:1 mux to implement function F

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Combinational vs. Sequential Circuits

- **Combinational**
  - Output depends only on current inputs

- **Sequential**
  - Output depends on current inputs plus past history
  - Includes memory

![Combinational Circuit](image)

![Sequential Circuit](image)
Sequential Circuits

- Outputs depend on inputs and state variables

- The state variables embody the past
  - Storage elements hold the state variables

- A clock periodically advances the circuit
Bistable Element

- Basic storage element
- Inverters with outputs connected to inputs

What does the circuit do?
Bistable = Two Stable States

- Bistable element stores a “given” value indefinitely (as long as powered)

How to change the stored value?
Revisit NOR and NAND Gates

• Null elements for NOR and NAND

• Implement NOT with NAND / NOR
S-R Latch (Set-Reset Latch)

Set (S) and Reset (R) inputs allow (re)setting stored value

$Q_{next}$ is new value of $Q$ when R or S changes

**Truth table?**

<table>
<thead>
<tr>
<th>S</th>
<th>R</th>
<th>$Q_{next}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
S-R Latch (Set-Reset Latch)

Boolean expression for \( Q_{\text{next}} \) in terms of \( R, S, \) and \( Q \):

\[
Q_{\text{next}} = (R + QN)' \\
= (R + (S + Q)')' \\
= R' \cdot (S + Q) \\
= R' \cdot S + R' \cdot Q
\]

When \( S=0 \) and \( R=0 \), it holds (latches) its previous state.

<table>
<thead>
<tr>
<th>( S )</th>
<th>( R )</th>
<th>( R' \cdot S )</th>
<th>( R' \cdot Q )</th>
<th>( Q_{\text{next}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Q</td>
<td>Q</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Q</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Instability (Avoid SR=11!)

• **SR=11 may create theoretical instability**
  1. When SR=11 $\rightarrow$ Q=QN=0
  2. If inputs subsequently change to SR=00, output may be undefined

```
 R  (RESET)          Q
    \
 S  (SET)  QN
     \
      Q
```
D Latch

- **D latch**: builds on **S-R latch** where **S** and **R** cannot be both 1
  - Output “follows” input

- **D latch captures input data (what to set)** when certain condition holds (when to set)

- **Operates in two modes**
  - Open (or transparent): input flows through to output
  - Closed (or opaque): output does not change
D Latch

- When C is enabled, Q output follows D input
- When C is disabled, Q output retains last state

<table>
<thead>
<tr>
<th>C</th>
<th>D</th>
<th>R</th>
<th>S</th>
<th>Q</th>
<th>QN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>Last Q</td>
<td>Last QN</td>
</tr>
</tbody>
</table>

Lecture 6: 13
Multi-bit Latch

- Simultaneously latches multiple bits
- “Latch” may refer to 1 bit latch or multi-bit one
Clock

• An input to a sequential circuit that changes output and state values at a predetermined rate

• Triggering edge: Transition of the clock (L→H or H→L) that captures input data
  – We mostly use positive edge as triggering edge in this class
  – Clock tick: Occurrence of a triggering edge
Clock Period and Frequency

• Clock Period (cycle time): Time between successive transitions in the same direction (L→H or H→L)
  – e.g., 1ms, 2ns, 250ps
• Clock Frequency: 1/period
  – e.g., 1kHz, 500MHz, 4GHz
D Latch Timing

D Latch

D

CLK

Q

CLK (open) (closed) (open) (closed)

D

Q
Flip-Flop

- **Samples input on** *triggering edge* of clock
  - Rising edge \(\rightarrow\) *positive edge-triggered flip-flop*
  - Falling edge \(\rightarrow\) *negative edge-triggered flip-flop*

- **D flip-flop:** Two D latches back-to-back
D Flip-Flop

- Copies D to Q on the rising edge of the clock
D Flip-Flop Timing

CLKL₁ (L₁ open) (L₁ closed) (L₁ open) (L₁ closed)
CLKL₂ (L₂ closed) (L₂ open) (L₂ closed) (L₂ open)

D Q₁ Q

Lecture 6: 20
Register

- Collection of FFs operating off common clock
- A single D flip-flop is a 1-bit register

![Diagram of a 4-bit register with D flip-flops connected to a common clock signal. The diagram shows the input (D) and output (Q) terminals for each flip-flop. The clock (CLK) signal is connected to all flip-flops.]
Before Next Class

- H&H 4.1-4.5 (skip VHDL parts), 5.4

Next Time

Counters
Shift Registers
Verilog