CSE140: Components and Design Techniques for Digital Systems

Sequential Circuit Introduction
Latches and Flip-Flops

Tajana Simunic Rosing

Sources: TSR, Katz, Boriello & Vahid
Where we are now….

• What we covered last time:
  – ALUs and other combinational circuits
• What we’ll do next:
  – Sequential circuits and memory
• Upcoming deadlines:
  – ZyBook today: Sec 4.1-2; Thursday: 4.3-6
  – HW#4 assigned, due next Tuesday
  – Quiz #4 on Thursday!
• Midterm #2
  – Coming up in 16 days – generally harder than Midterm#1
  – Cumulative

Sources: TSR, Katz, Boriello & Vahid
What is a sequential circuit?

A circuit whose output depends on current inputs and past outputs

A circuit with memory

\[
\begin{align*}
S_i & = f_i(S^t, X) \\
S_{i+1} & = g_i(S^t, X)
\end{align*}
\]
Why do we need circuits with ‘memory’?

- Circuits with memory can be used to store data
- Systems have circuits that run a sequence of tasks

Memory Hierarchy

- Registers
- Cache
- Main Memory
- SSD
- Hard disk

Sources: TSR, Katz, Boriello & Vahid
Simplest memory element

SRAM = Static RAM = 6T

"remember"

"data"

"load"

"stored value"

Sources: TSR, Katz, Boriello & Vahid
Flight attendant call button

• Flight attendant call button
  – Press call: light turns on
    • *Stays on* after button released
  – Press cancel: light turns off
  – Logic circuit to implement this?

• SR latch implementation
  – Call=1 : sets Q to 1 and keeps it at 1
  – Cancel=1 : resets Q to 0

Sources: TSR, Katz, Boriello & Vahid
SR Latch Analysis

- $S = 1$, $R = 0$:
  then $Q = 1$ and $\overline{Q} = 0$
  \begin{itemize}
  \item \textit{set} \hspace{1cm} \textit{store 1}
  \end{itemize}

- $S = 0$, $R = 1$:
  then $Q = 1$ and $\overline{Q} = 0$
  \begin{itemize}
  \item \textit{reset} \hspace{1cm} \textit{store 0}
  \end{itemize}

Sources: TSR, Katz, Boriello & Vahid
SR Latch Analysis

– $S = 0$, $R = 0$: then $Q = Q_{prev}$
– Memory!

$Q_{prev} = 0$

$Q_{prev} = 1$

– $S = 1$, $R = 1$: then $Q = 0$, $\bar{Q} = 0$
– Invalid State
$\bar{Q} \neq \text{NOT } Q$

Sources: TSR, Katz, Boriello & Vahid
What if a kid presses both call and cancel & then releases them?

- If $S=1$ and $R=1$ at the same time and then released, $Q=\text{?}$
  - Can also occur also due to different delays of different paths
  - $Q$ may oscillate and eventually settle to 1 or 0 due to diff. path delay

<table>
<thead>
<tr>
<th>$S$</th>
<th>$R$</th>
<th>$Q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>hold</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>not allowed</td>
</tr>
</tbody>
</table>

Sources: TSR, Katz, Boriello & Vahid
SR Latch Symbol

• SR stands for Set/Reset Latch
  – Stores one bit of state ($Q$)

• Control what value is being stored with $S$, $R$ inputs
  – **Set:** Make the output 1
    
    ($S = 1$, $R = 0$, $Q = 1$)

  – **Reset:** Make the output 0
    
    ($S = 0$, $R = 1$, $Q = 0$)

  – **Hold:** Keep data stored
    
    ($S = 0$, $R = 0$, $Q = Q_{\text{previous}}$)
SR Latch Characteristic Equation

To analyze, break the feedback path

<table>
<thead>
<tr>
<th>S</th>
<th>R</th>
<th>Q(t)</th>
<th>Q(t+Δ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</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>0</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>1</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>X</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>X</td>
</tr>
</tbody>
</table>

characteristic equation

\[ Q(t+\Delta) = S + R' \cdot Q(t) \]

\[ \overline{Q(t+\Delta)} = (S + \overline{Q}) \cdot \overline{R} \]
Avoiding S=R=1 Part 1: Level-Sensitive SR Latch

- Add input “C” = clock
  - Change C to 1 only after S and R are stable
  - C is usually a clock (CLK)

SR latch circuit diagram:

\[ Q \]
\[ S \]
\[ C_{clk} \]
\[ R \]
\[ S1 \]
\[ R1 \]

Conditions:
- If \( C_{clk} = 1 \)
  - \( S1 = S \) & \( R1 = R \)
- If \( C_{clk} = 0 \)
  - \( S1 = \phi = R1 \)
  - SR latch is hold (mem)

Sources: TSR, Katz, Boriello & Vahid
- **Clock** -- Pulsing signal for enabling latches; ticks like a clock
- **Synchronous** circuit: sequential circuit with a clock

- **Clock period**: time between pulse starts
  - Above signal: period = 20 ns
- **Clock cycle**: one such time interval
  - Above signal shows 3.5 clock cycles
- **Clock duty cycle**: time clock is high
  - 50% in this case
- **Clock frequency**: 1/period
  - Above: freq = 1 / 20ns = 50MHz;

<table>
<thead>
<tr>
<th>Freq</th>
<th>Period</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 GHz</td>
<td>0.01 ns</td>
</tr>
<tr>
<td>10 GHz</td>
<td>0.1 ns</td>
</tr>
<tr>
<td>1 GHz</td>
<td>1 ns</td>
</tr>
<tr>
<td>100 MHz</td>
<td>10 ns</td>
</tr>
<tr>
<td>10 MHz</td>
<td>100 ns</td>
</tr>
</tbody>
</table>
Clock question

The clock shown in the waveform below has:

A. Clock period of 4ns with 250MHz frequency
B. Clock duty cycle 75%
C. Clock period of 1ns with 1GHz frequency
D. A. & B.
E. None of the above

\[ \frac{1}{4\text{ns}} = 0.25 \times 10^9 \]

\checkmark A. Clock period of 4ns with 250MHz frequency
\checkmark B. Clock duty cycle 75%
\xmark C. Clock period of 1ns with 1GHz frequency
\xmark D. A. & B.
\xmark E. None of the above
Avoiding S=R=1 Part 2: Level-Sensitive D Latch

- SR latch requires careful design so SR=11 never occurs
- D latch helps by inserting the inverter between S & R inputs
  - Inserted inverter ensures R is always the opposite of S when C=1
D Latch Truth Table

<table>
<thead>
<tr>
<th>CLK</th>
<th>D</th>
<th>$\overline{D}$</th>
<th>S</th>
<th>R</th>
<th>Q</th>
<th>$\overline{Q}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>X</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$\overline{Q_{\text{prev}}}$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>$\overline{Q_{\text{prev}}}$</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>$\overline{Q_{\text{prev}}}$</td>
</tr>
</tbody>
</table>

$Q = D$

Sources: TSR, Katz, Boriello & Vahid
D Latch Summary

- Two inputs: \( CLK, D \)
  - \( CLK \): controls *when* the output changes
  - \( D \) (the data input): controls *what* the output changes to

- Function
  - When \( CLK = 1 \),
    - \( D \) passes through to \( Q \) (*transparent*)
  - When \( CLK = 0 \),
    - \( Q \) holds its previous value (*opaque*)

- (Mostly) avoids invalid case \( Q = Q' \)

Sources: TSR, Katz, Boriello & Vahid
Level-Sensitive D Latches

Assume that data in all latches is initially 0. Input Y=1 and Clk transitions from 0->1. When Clk=0 again, the stored values in latches are:

A. Q1=1, Q2=0, Q3=0, Q4=0 for both clock A & B
B. Q1=1, Q2=1, Q3=1, Q4=1 for clock A
   Q1=1, Q2=0, Q3=0, Q4=0 for clock B
C. Q1=1, Q2=1, Q3=1, Q4=1 for both clocks
D. More information is needed to determine the answer
E. None of the above

Sources: TSR, Katz, Boriello & Vahid
**D Flip-Flop Design & Timing Diagram**

- **Flip-flop**: Bit storage that stores on the clock edge, not level
- Master-slave design: master loads when Clk=0, then slave when Clk=1

Sources: TSR, Katz, Boriello & Vahid
D Flip-Flop: Characteristic Equation

Characteristic Equation

\[ Q(t+1) = D(t) \]