Combinational Building Blocks

- Multiplexers
- Decoders
Multiplexer (Mux)

- Selects between one of $N$ inputs to connect to output
  - $\log_2 N$-bit required to select input – control input $S$

- Example:
  2:1 Mux (2 inputs to 1 output)
  - $N = 2$
  - $\log_2 2 = 1$ control bit required
### Multiplexer Implementations

#### Logic gates
- Sum-of-products form

<table>
<thead>
<tr>
<th>$S$</th>
<th>$D_1$</th>
<th>$D_0$</th>
<th>$Y$</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>1</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>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

#### Tristates
- For an N-input mux, use N tristates
  - Turn on exactly one to select the appropriate input
Logic using Multiplexers

- Using the mux as a lookup table
  - Zero outputs tied to GND
  - One output tied to VDD
Logic using Multiplexers

• Reducing the size of the mux

\[ Y = AB \]
Decoders

- $N$ inputs, $2^N$ outputs
- One-hot outputs: only one output HIGH at once

Example:

2:4 Decoder (2 inputs to 4 outputs)
- $A_i$ decimal value selects the corresponding output

<table>
<thead>
<tr>
<th>$A_1$</th>
<th>$A_0$</th>
<th>$Y_3$</th>
<th>$Y_2$</th>
<th>$Y_1$</th>
<th>$Y_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Decoder Implementation

A_1 \quad A_0

- Y_3
- Y_2
- Y_1
- Y_0
• OR minterms

\[ Y = AB + \overline{AB} \]

\[ = A \oplus B \]
• Delay between input change and output changing

• How to build fast circuits?
• **Propagation delay:** \( t_{pd} = \) max delay from input to final output

• **Contamination delay:** \( t_{cd} = \) min delay from input to initial output change

Note: Timing diagram shows a signal with a high and low and transition time as an ‘X’. Cross hatch indicates unknown/changing values.
Delay is caused by
- Capacitance and resistance in a circuit
- Speed of light limitation

Reasons why $t_{pd}$ and $t_{cd}$ may be different:
- Different rising and falling delays
- Multiple inputs and outputs, some of which are faster than others
- Circuits slow down when hot and speed up when cold
Critical (Long) & Short Paths

Critical (Long) Path: \( t_{pd} = 2t_{pd_{AND}} + t_{pd_{OR}} \)

Short Path: \( t_{cd} = t_{cd_{AND}} \)
Glitches

• When a single input change causes an output to change multiple times
• What happens when $A = 0$, $C = 1$, $B$ falls?

$Y = AB + BC$
Glitch Example (cont.)

Note: n1 is slower than n2 because of the extra inverter for B to go through.
Fixing the Glitch

Consensus term \( \bar{A}C \)

\[
Y = \bar{A}B + BC + \bar{A}C
\]

\[
\begin{array}{c|c|c|c|c}
A & B & C & Y \\
0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 \\
0 & 1 & 1 & 1 \\
1 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 \\
\end{array}
\]

\[
A = 0
B = 1 \rightarrow 0
C = 1
\]

\[
Y = 1
\]
Why Understand Glitches?

- Glitches shouldn’t cause problems because of **synchronous design** conventions (see Chapter 3)
- It’s important to **recognize** a glitch: in simulations or on oscilloscope
- Can’t get rid of all glitches – simultaneous transitions on multiple inputs can also cause glitches