Multiplexer / Demultiplexer (mux/demux)

- combinational logic used to combine/distribute $2^n$ inputs/outputs addressed by $n$ select lines into/from a single line

Eg. 4:1 MUX

Applications

1) sharing bus (= data line)

- Mux combines multiple inputs into a single data stream
- Demux splits the data stream into the orig. multiple signals

Also: can use 3-state logic
- cannot share usual 0/1 devices without confusion
- each device must have tristate buffer, which effectively removes all but one device from the bus

\[ A \xrightarrow{\text{3-state buffer}} X = A \xrightarrow{\text{3 states}} X \]
3 states:
\[
\begin{align*}
0/1 & \text{ logic} \\
Z & \text{ high Z (open circuit)}
\end{align*}
\]

2) Arbitrary logic implementation (cf. PLD)

\[ 2^n : 1 \text{ mux can represent arbitrary } n \text{-input func} \]

E.g. XOR

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>X</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>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

3) Encoding /decoding (encoder/decoder -another name for mux/demux)

- turn one form of digital info representation into another /and back

BCD ↔ binary
Gray ↔ decimal
Example
Priority encoder

Input

\[ V_{\text{ref}} \]

\[ V_m \]

\[ \text{output from parallel ADC} \]

\[ \begin{array}{c|c}
X_2 & Y_1 \ Y_0 \\
000 & 00 \\
001 & 01 \\
011 & 10 \\
111 & 11 \\
\end{array} \]

Need to convert to binary
(= priority encoder)

Implementation

with two 8:1 muxes

\[ X_2 \ X_1 \ X_0 \]

\[ Y_1 \ Y_0 \]

\[ X_2 \ X_1 \ X_0 \]

\[ Y_0 \]

Priority decoder?
Eg strength bar display

\[ \text{LED} \]

\[ \text{weak} \]

\[ \text{strong} \]
Flip-flops (contd.)
so far discussed RS flip-flop

D flip-flop (data - 1 bit memory)

```
D Q
0 0
1 1
```

JK flip-flop

```
J K
0 0
0 1
1 0
1 1
```

T flip-flop (toggle)

```
EN Q
Qn+1 = Qn when EN = 1
```

- any type of FF (+ combinational gates) can be converted into any other type

```
e.g. D → T
```