Today

- Timing in DesignWorks
  - Generating Useful Output
  - Gate Delay Properties
- Notes on Multiplexers as general-purpose logic
- Demultiplexers versus Decoders
- BCD to 7-segment display decoder example
- Hardware Demonstration

Timing in DesignWorks: Generating useful output

- Input Vector
- Waveform Output
- Refer to tips page for How-To
Timing in DesignWorks: Gate Delay Properties

- Adjust the overall delay time of the primitive device
- Show all fields in the device attributes
- Set Delay.Dev to an integer delay value
- Delay.Dev.(Typ/Min/Max) are information storage only and not interpreted by DesignWorks
- Delay.Pin is not the device delay
- 1 is the default
- 0 possible but hard to reason about

Notes on Multiplexers as general-purpose logic

- A \(2^{n-1}:1\) multiplexer can implement any function of \(n\) variables
- with \(n-1\) variables used as control inputs and
- the data inputs tied to the last variable or its complement
- Example:
  \[ F(A,B,C) = m_0 + m_2 + m_6 + m_7 \\
  = A'B'C' + A'BC' + ABC' + ABC \\
  = A'B(C') + A'B(C') + AB'(0) + AB'(1) \]
Notes on Multiplexers as general-purpose logic (cont’d)

- Sometimes it can be reduced even further to a $2^{n-k}:1$ multiplexer ($k>1$) but there is no standard method to determine this.

- $F(A,B,C) = m_0 + m_2 + m_6 + m_7$
  - $= A'B'C' + A'BC' + ABC' + ABC$
  - $= A'B'(C') + A'B(C') + AB'(0) + AB(1)$
  - $= A'(C') + A(B)$

Demultiplexers versus Decoders

- Demultiplexers are simply the most general class of Decoder with a full AND array.
- In general
  - any form of partial AND array
  - mapping of $m$ inputs to $n$ outputs where $n > m$. 
BCD to 7-segment display controller

- Example will be covered again and in greater detail in lecture
- Understanding the problem
  - input is a 4 bit bcd digit (A, B, C, D)
  - output is the control signals for the display (7 outputs C0 - C6)
- Block diagram:

Formalize the problem

- Truth table
  - show don't cares
- Implementation procedure
  - minimization using K-maps
  - map to hardware of some type

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>C0</th>
<th>C1</th>
<th>C2</th>
<th>C3</th>
<th>C4</th>
<th>C5</th>
<th>C6</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<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>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<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>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Implementation as minimized SOP

- 15 unique product terms when minimized individually

```
A B C D
0 1 1 1
1 0 1 1
1 1 0 1
1 1 1 0
1 1 1 1
```

Implementation as minimized SOP (cont'd)

- Can do better than minimized
  - 9 unique product terms (instead of 15)
  - Share terms among outputs, good for PLAs
  - Each output not necessarily in minimized form

```
A B C D
0 1 1 1
1 0 1 1
1 1 0 1
1 1 1 0
1 1 1 1
```

C0 = A + B D + C + B' D'
C1 = C' D' + C D + B'
C2 = B + C' + D
C3 = B' D' + C D' + B C' D + B' C
C4 = B' D' + C D'
C5 = A + C' D' + B D' + B' C
C6 = A + C D' + B C' + B' C

C0 = A + B D + C + B' D'
C1 = C' D' + C D + B'
C2 = B + C' + D
C3 = B' D' + C D' + B C' D + B' C
C4 = B' D' + C D'
C5 = A + C' D' + B D' + B' C
C6 = A + C D' + B C' + B' C

C0 = B' C' D + C D + B' D' + B C D' + A
C1 = C' D' + C D + B'
C2 = B + C' + D
C3 = B' D' + C D' + B C' D + B' C
C4 = B' D' + C D'
C5 = A + C' D' + B D' + B' C
C6 = A + C D' + B C' + B' C

C0 = B' C' D + C D + B' D' + B C D' + A
C1 = C' D' + C D + B'
C2 = B + C' + D
C3 = B' D' + C D' + B C' D + B' C
C4 = B' D' + C D'
C5 = A + C' D' + B D' + B' C
C6 = A + C D' + B C' + B' C
Hardware Demonstration: Protoboard Layout

- Power and ground
- Distribution Channels
- Pin Connections

Hardware Demonstration: TTL Parts

- Obfuscated Technology (gates-on-a-chip)
- Motorola Specification Sheets
- The 4th Floor Beast
- Local Availability in EE1