Announcements

• Homework 6 due today
Agenda

• Last time:
  – Binary Adders and Subtracters (5.1, 5.1.1)
  – Carry Lookahead Adders (5.1.2, 5.1.3)

• This time:
  – Decimal Adders (5.2)
  – Comparators (5.3)
  – Decoders (5.4)
  – Encoders (5.5)
  – Multiplexers (5.6)

• After introducing all the MSI components, we will go back and talk about how to use Decoders and Multiplexers for Logic Design.
# Decimal Adders

8421 weighted coding scheme or BCD Code

<table>
<thead>
<tr>
<th>Decimal Digit</th>
<th>BCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0000</td>
</tr>
<tr>
<td>1</td>
<td>0001</td>
</tr>
<tr>
<td>2</td>
<td>0010</td>
</tr>
<tr>
<td>3</td>
<td>0011</td>
</tr>
<tr>
<td>4</td>
<td>0100</td>
</tr>
<tr>
<td>5</td>
<td>0101</td>
</tr>
<tr>
<td>6</td>
<td>0110</td>
</tr>
<tr>
<td>7</td>
<td>0111</td>
</tr>
<tr>
<td>8</td>
<td>1000</td>
</tr>
<tr>
<td>9</td>
<td>1001</td>
</tr>
</tbody>
</table>

Forbidden codes: 1010, 1011, 1100, 1101, 1110, 1111
Decimal Adder

• Inputs: $A_3A_2A_1A_0$, $B_3B_2B_1B_0$, $C_{in}$ from previous decade.
• Output: $C_{out}$ (carry to next decade), $Z_3Z_2Z_1Z_0$.
• Idea: Perform regular binary addition and then apply a corrective procedure.
## Comparing Binary and BCD Sums

<table>
<thead>
<tr>
<th>Decimal Sum</th>
<th>K</th>
<th>$P_3$</th>
<th>$P_2$</th>
<th>$P_1$</th>
<th>$P_0$</th>
<th>$C_{out}$</th>
<th>$Z_3$</th>
<th>$Z_2$</th>
<th>$Z_1$</th>
<th>$Z_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Same</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>0</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>1</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>1</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>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>0</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>
</tr>
</tbody>
</table>

*C_{out} is set to 0*
Decimal Adder

• No correction needed when the decimal sum is between 0-9.
• Must apply a correction when the sum is between 10-19.
• Case 1:
  – 16-19: K is set to 1. Add binary quantity 0110 to $P_3P_2P_1P_0$.
  – 10-15: $KP_3P_2P_1P_0$ are set to 01010, 01011, . . . , 01111. Need to add 6. Use a K-map to obtain a Boolean expression to detect these six binary combinations.
A single-decade BCD Adder
Comparators

• Compare the magnitude of two binary numbers for the purpose of establishing whether one is greater than, equal to, or less than the other.

• A comparator makes use of a cascade connection of identical subnetworks similar to the case of the parallel adder.
Comparators

- Consider two n-bit binary numbers:
  \[ A = A_{n-1} \cdots A_iA_{i-1} \cdots A_1A_0 \]
  \[ B = B_{n-1} \cdots B_iB_{i-1} \cdots B_1B_0 \]
- Assume \( A_i, B_i \) are entering the subnetwork and that the binary numbers are analyzed from right to left.
- Subnetwork is called a 1-bit comparator.
Comparators

- 3 conditions describing the relative magnitudes of $A_{i-1} \cdots A_1A_0$, $B_{i-1} \cdots B_1B_0$
- $G_i = 1$ denotes $A_{i-1} \cdots A_1A_0 > B_{i-1} \cdots B_1B_0$
- $E_i = 1$ denotes $A_{i-1} \cdots A_1A_0 = B_{i-1} \cdots B_1B_0$
- $L_i = 1$ denotes $A_{i-1} \cdots A_1A_0 < B_{i-1} \cdots B_1B_0$
- 1-bit comparator is a 5-input 3-output network
Comparators

- Rules:
  - If $A_i = 0$, $B_i = 1$ then $L_i = 1$
  - If $A_i = 1$, $B_i = 0$ then $G_i = 1$
  - If $A_i = B_i$ and $L_{i-1} = 1$ then $L_i = 1$
  - If $A_i = B_i$ and $G_{i-1} = 1$ then $G_i = 1$
  - If $A_i = B_i$ and $E_{i-1} = 1$ then $E_i = 1$

- Can use this to construct a truth table.
Comparators

- Minimal Sum Boolean Expressions:

\[ G_{i+1} = A_i \overline{B_i} + A_i G_i + \overline{B_i} G_i \]

\[ E_{i+1} = A_i \overline{B_i} E_i + A_i B_i E_i \]

\[ L_{i+1} = A_i B_i + B_i L_i + \overline{A_i} L_i \]
Comparators
Comparators
Decoder

• Digital information represented in some binary form must be converted into some alternate binary form.

• $n$ to $2^n$-line decoder.

• Only one of the $2^n$ output lines responds, with a logic-1, to a given input combination of values on its $n$-input lines.
Realization

Logic Diagram

Truth Table

Symbol
Decoder

• Input combinations can be regarded as binary numbers with the consequences that the j-th output line is at logic-1 for \( j = 0, 1, \ldots, 7 \) only when input combination \( j \) is applied.
Decoders with an Enable Input

Logic Diagram

Truth Table

Symbol
Decoders with enable inputs

• When disabled, all outputs of the decoder can either be at logic-0 or logic-1.
• Enable input provides the decoder with additional flexibility. Idea: data is applied to the enable input.
• Process is known as demultiplexing.

If $x_0 = 0, x_1 = 0$ then data appears on line $z_0$.

• Enable inputs are useful when constructing larger decoders from smaller decoders.
Constructing Larger Decoders

Figure 5.29 A 4-to-16-line decoder constructed from 2-to-4-line decoders.
Encoders

- Encoders provide for the conversion of binary information from one form to another.
- Encoders are essentially the inverse of decoders.
- $2^n$-to-$n$-line encoder in which an assertive logic value on one of its $2^n$-input lines causes the corresponding binary code to appear at the output lines.

<table>
<thead>
<tr>
<th>$x_0$</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
<th>$x_5$</th>
<th>$x_6$</th>
<th>$x_7$</th>
<th>$z_0$</th>
<th>$z_1$</th>
<th>$z_2$</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>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</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>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<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>
</tr>
<tr>
<td>1</td>
<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>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Encoders

• Equations for 8-to-3-line encoder:
  
  \[ z_0 = x_1 + x_3 + x_5 + x_7 \]
  \[ z_1 = x_2 + x_3 + x_6 + x_7 \]
  \[ z_2 = x_4 + x_5 + x_6 + x_7 \]

• In general, the Boolean expression for the output \( z_i \) is the sum of each input \( x_j \) in which the binary representation of \( j \) has a 1 in the \( 2^i \)-bit position.

---

**Figure 5.30** A \( 2^n \)-to-\( n \)-line encoder symbol.

**Figure 5.31** An 8-to-3-line encoder.
Priority Encoder

• The assumption that at most a single input to the encoder is asserted at any time is significant in its operation.
  – Example: Both $x_3 \ (11)$ and $x_5 \ (101)$ are asserted. What is the output?
    – 111 ($x_7$)

• Priority Encoder:
  – A priority scheme is assigned to the input lines so that whenever more than one input line is asserted at any time, the output is determined by the input line having the highest priority.
Priority Encoder

The output is determined by the asserted input having the highest index. $x_i$ has higher priority than $x_j$ if $i > j$.

“Valid” indicates that at least one input line is asserted. This distinguishes the situation that no input line is asserted from when the $x_0$ input line is asserted, since in both cases $z_2z_1z_0 = 000$. 

---

**Table 5.5** Compressed truth table for an 8-to-3 line priority encoder

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_0$</td>
<td>$z_2$</td>
</tr>
<tr>
<td>$x_1$</td>
<td>$z_1$</td>
</tr>
<tr>
<td>$x_2$</td>
<td>$z_0$</td>
</tr>
<tr>
<td>$x_3$</td>
<td>Valid</td>
</tr>
<tr>
<td>$x_4$</td>
<td></td>
</tr>
<tr>
<td>$x_5$</td>
<td></td>
</tr>
<tr>
<td>$x_6$</td>
<td></td>
</tr>
<tr>
<td>$x_7$</td>
<td></td>
</tr>
</tbody>
</table>
Multiplexer

• Also called data selectors.
• Basic function: select one of its $2^n$ data input lines and place the corresponding information onto a single output line.
• $n$ input bits needed to specify which input line is to be selected.
  – Place binary code for a desired data input line onto its $n$ select input lines.
Realization of 4-to-1 line multiplexerer

Logic Diagram

Truth Table

Symbol
Realization of 4-to-1 line multiplexer

• Alternate description:

\[ f = I_0 S_1 \bar{S}_0 + I_1 \bar{S}_1 S_0 + I_2 S_1 \bar{S}_0 + I_3 S_1 S_2 \] \( E \)

<table>
<thead>
<tr>
<th>( E )</th>
<th>( S_1 )</th>
<th>( S_0 )</th>
<th>( f )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( \times )</td>
<td>( \times )</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>( I_0 )</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>( I_1 )</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>( I_2 )</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>( I_3 )</td>
</tr>
</tbody>
</table>

• Algebraic description of multiplexer:

\[ f = (I_0 S_1 \bar{S}_0 + I_1 \bar{S}_1 S_0 + I_2 S_1 \bar{S}_0 + I_3 S_1 S_2)E \]
Building a Large Multiplexer
Multiplexers

• One of the primary applications of multiplexers is to provide for the transmission of information from several sources over a single path.
• This process is known as multiplexing.
• Demultiplexer = decoder with an enable input.
Multiplexer/Demultiplexer for information transmission

Figure 5.35 A multiplexer/demultiplexer arrangement for information transmission.