CSE 140 Lecture 11
Standard Combinational Modules

CK Cheng
CSE Dept.
UC San Diego
Part III - Standard Combinational Modules

• Introduction
• Decoder
  – Behavior, Logic, Usage
• Encoder
• Multiplexer (Mux)
  – Behavior, Logic, Usage
• Demultiplexer (DeMux)
Part III - Standard Combinational Modules

**Signal Transport**
- Decoder: Decode address
- Encoder: Encode address
- Multiplexer (Mux): Select data by address
- Demultiplexer (DeMux): Direct data by address
- Shifter: Shift bit location

**Data Operator**
- Adder: Add two binary numbers
- Multiplier: Multiply two binary numbers
Interconnect: Decoder, Encoder, Mux, DeMux

Decoder: Decode the address to assert the addressed device
Mux: Select the inputs according to the index addressed by the control signals
Interconnect: Decoder, Encoder, Mux, DeMux

Processors

Memory Bank

Arbiter
1. Decoder

- Definition
- Logic Diagram
- Application (Universal Set)
- Tree of Decoders
Decoder: Definition

A. A device that decodes
B. An electronic device that converts signals from one form to another
C. A machine that converts a coded text into ordinary language
D. A device or program that translates encoded data into its original format
E. All of the above
Decoder Definition: A digital module that converts a binary *address* to the assertion of the addressed *device*

n inputs

n= 3

2^n outputs

2^3 = 8

n to 2^n decoder function:

\[ y_i = 1 \text{ if } E = 1 \& (I_2, I_1, I_0) = i \]

\[ y_i = 0 \text{ otherwise} \]
1. Decoder: Definition

- \( N \) inputs, \( 2^N \) outputs
- One-hot outputs: only one output HIGH at most

<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>
1. Decoder: Definition

Example: A 3-input decoder has how many outputs?

A. 2 outputs
B. 4 outputs
C. 8 outputs
D. 10 outputs

↓^E
Example: For a 3-input decoder, suppose $(E, I_2, I_1, I_0) = (1, 0, 0, 0)$, then $(y_7, y_6, \ldots, y_0)$ is equal to:

A. (00000000)
B. (00000001)
C. (00000010)
D. (01000000)
E. (10000000)
Decoder: Logic Diagram (Inside a decoder)

\[ y_0 = 1 \text{ if } (A_1, A_0) = (0,0) \& E = 1 \]

\[ y_i = m_i E \]

\[ y_3 = A_1 A_0 E \]
1. Decoder: Definition

PI Q: What is the output $Y_{3:0}$ of the 2:4 decoder for $(A_1, A_0) = (1,0)$?

A. (1, 1, 0, 0 )
B. (1, 0, 1, 1)
C. (0, 0, 1, 0)
D. (0, 1, 0, 0)
Decoder Application: universal set \{Decoder, OR\}

Example:
Implement the following functions with a 3-input decoder and OR gates.

i)  \( f_1(a,b,c) = \Sigma m(1,2,4) \)

ii) \( f_2(a,b,c) = \Sigma m(2,3) \),

iii) \( f_3(a,b,c) = \Sigma m(0,5,6) \)
Decoder Application: universal set \{Decoder, OR\}

Decoder produces minterms when \(E=1\).
We can use an OR gate to collect the minterms to cover the On-set.
For the Don’t Care-Set, we can just ignore the terms.
Decoder Application: universal set \{\text{Decoder, OR}\}

Example: Implement functions
i) \(f_1(a,b,c) = \Sigma m(1,2,4) + \Sigma d(0,5)\),
ii) \(f_2(a,b,c) = \Sigma m(2,3) + \Sigma d(1,4)\),
iii) \(f_3(a,b,c) = \Sigma m(0,5,6)\)

with a 3-input decoder and OR gates.
Decoders

- OR minterms

\[ Y = AB + \overline{AB} = A \oplus B \]

\[ E=1 \]

\[ \begin{array}{cccc}
A & B & 11 & 10 \\
01 & 00 & & \\
\end{array} \]

Minterm

\[ \begin{array}{cccc}
AB & \overline{AB} & \overline{AB} & \overline{AB} \\
\end{array} \]
Tree of Decoders: Scale up the size of the decoders using a tree structure

Implement a $4-2^4$ decoder with $3-2^3$ decoders.
Implement a $6-2^6$ decoder with $3-2^3$ decoders.
Construction: A four variable switching function $f(a,b,c,d)$ can be implemented using which of the following?

A. 1:2 decoders and OR gates
B. 2:4 decoders and OR gates
C. 3:8 decoders and OR gates
D. All of the above
E. None of the above
2. Encoder

• Definition
• Logic Diagram
• Priority Encoder
Definition of Encoder

A. Any program, circuit or algorithm which encodes
B. In digital audio technology, an encoder is a program that converts an audio WAV file into an MP3 file
C. A device that convert a message from plain text into code
D. A circuit that is used to convert between digital video and analog video
E. All of the above
Encoder Definition: A digital module that converts the assertion of a **device** to the binary **address** of the device.

**Encoder Description:**

At most one $I_i = 1$.  
$(y_{n-1},...,y_0) = i$ if $I_i = 1 \& E = 1$  
$(y_{n-1},...,y_0) = 0$ otherwise.  
$A = 1$ if $E = 1$ and one $i$ s.t. $I_i = 1$  
$A = 0$ otherwise.
Encoder: Logic Diagram
Priority Encoder:

I₀  I₁  I₂  I₃

E  →  0  →  y₀
    
1  →  1  →  y₁

E₀  Gₛ
Priority Encoder: Definition

Description: Input \((I_{2^{n-1}}, \ldots, I_0)\), Output \((y_{n-1}, \ldots, y_0)\)

\[(y_{n-1}, \ldots, y_0) = i \text{ if } I_i = 1 \& E = 1 \& I_k = 0 \text{ for all } k > i \text{ (high bit priority)} \text{ or} \]
\[\text{for all } k < i \text{ (low bit priority).} \]

\[E_o = 1 \text{ if } E = 1 \& I_i = 0 \text{ for all } i, \]
\[G_s = 1 \text{ if } E = 1 \& \exists i \text{ s.t. } I_i = 1. \]

\((G_s \text{ is like } A, \text{ and } E_o \text{ passes on enable}).\)
Priority Encoder: Implement a 32-input priority encoder with 8 input priority encoders (high bit priority).