Logic Components & QM Logic Minimization
Outline

• Karnaugh map with >= 5 variable

• Study of Components
  – Multiplexor, Decoder
  – Logic Implementation Using MUX & Decoder
  – Mux: 7 Segment Display
  – 4 Bit Adder

• Quine-McCluskey (QM) Logic Minimization

• More Examples
Multiple-Output Circuits

• Many circuits have more than one output
• Can give each a separate circuit, or can share gates

• Ex: $F = ab + c'$, $G = ab + bc$

Option 1: Separate circuits
Option 2: Shared gates
Decoder

• Reception counter: When you reach a Academic Institute
  – Receptionist Ask: Which Dept to Go?
  – Receptionist Redirect you to some building according to your Answer.

• Decoder: knows what to do with this: Decode

• N input: $2^N$ output

• Memory Addressing
  – Address to a particular location
Decoders

- **Decoder**: Popular combinational logic building block, in addition to logic gates
  - Converts input binary number to one high output
- 2-input decoder: four possible input binary numbers
  - So has four outputs, one for each possible input binary number

<table>
<thead>
<tr>
<th>i0</th>
<th>d0</th>
<th>d1</th>
<th>d2</th>
<th>d3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<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>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>i0</th>
<th>d0</th>
<th>d1</th>
<th>d2</th>
<th>d3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</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>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Decoders and Muxes

• Internal design
  – AND gate for each output to detect input combination

• Decoder with enable e
  – Outputs all 0 if e=0, Regular behavior if e=1

• n-input decoder: \(2^n\) outputs

Covers All Minterms
4-to-16 Decoder using two 3-to-8 Decoders

Diagram:

- Inputs: x, y, z, w
- Outputs: D0 to D7, D8 to D15
- Two 3-to-8 decoders
- Connections:
  - x, y, z to the first decoder
  - w to the second decoder
  - Outputs of the decoders are connected to the outputs D0 to D7, D8 to D15
Boolean Function Implementation using Decoders

• Using a n-to-2n decoder and OR gates any functions of n variables can be implemented.

• Example:

  \[ S(x, y, z) = \Sigma(1, 2, 4, 7) \], \[ C(x, y, z) = \Sigma(3, 5, 6, 7) \]

• Functions S and C can be implemented using a 3-to-8 decoder and two 4-input OR gates
Implementation of S and C
Configurable Function using Decoder & Memory

Any Function can be implemented

8 row x 1 col memory

Function depend on Memory Elements, Direct correspondence to Truth Table
Multiplexor (Mux)

- Mux: Another popular combinational building block
  - Routes one of its $N$ data inputs to its one output, based on binary value of select inputs
    - 4 input mux $\rightarrow$ needs 2 select inputs to indicate which input to route through
    - 8 input mux $\rightarrow$ 3 select inputs
    - $N$ inputs $\rightarrow$ $\log_2(N)$ selects
  - Like a railyard switch
Mux Internal Design

2x1 mux

\[ i0 \times 1 = i0 \]

\[ (0 + i0 = i0) \]
Mux Internal Design

4x1 mux

Covers All Minterms
Muxes Commonly Together -- N-bit

Mux

• Ex: Two 4-bit inputs, A (a3 a2 a1 a0), and B (b3 b2 b1 b0)
  – 4-bit 2x1 mux (just four 2x1 muxes sharing a select line) can select between A or B
Multiplexed 7 Segment Decoder

MUX → 7 Segment Decoder → DEMUX

HR1 → 4 → MUX → 4 → 7 Segment Decoder → 7 → DEMUX

HR2 → 4 → MUX → 4 → 7 Segment Decoder → 7 → DEMUX

MIN1 → 4 → MUX → 4 → 7 Segment Decoder → 7 → DEMUX

MIN2 → 4 → MUX → 4 → 7 Segment Decoder → 7 → DEMUX

2 bit counter Running At KhZ

HOUR

MIN
Implementing logic Function using MUX

\[ F(A, B) = \sum m(0, 2) \]

4x1 mux

[Diagram of a 4x1 multiplexer]