Lect 10

Combinational Block design

CS221: Digital Design

Dr. A. Sahu
Dept of Comp. Sc. & Engg.
Indian Institute of Technology Guwahati

Outline

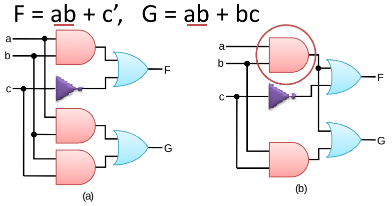
- Combinational Block
- Encoder, Priority Encoder
  - Decoder: N to 2^N, Encoder: 2^N to N
  - Decoder : Output selector
- Multiplexor: The input selector……
  - 2^N to 1, N select line
- Adder, Substractor, BCD Adder
- Binary Multiplier
- Other Encoders

Study of Components

- Decoder, Encoder
- Multiplexor
- Logic Implementation Using MUX & Decoder
- Mux: 7 Segment Display
- 4 Bit Adder
- N- Bit Adder

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



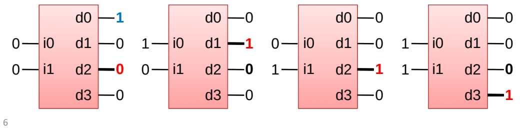
Option 1: Separate circuits Option 2: Shared gates

Decoder

- Reception counter : When you reach a Academic Institute
  - Receptionist Ask: Which Dept to Go ?
  - Customer : CSE
  - Receptionist Redirect you to some building according to your Answer. == > Go to Core II
- Decoder : knows what to do with this: Decode
- Digital Case: == > N input: 2^N output
- Memory Addressing
  - Address to a particular location

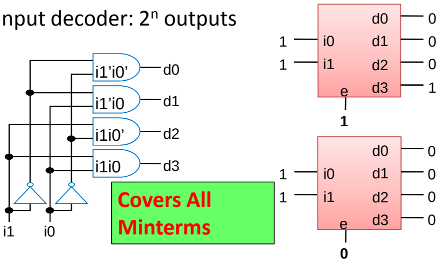
Decoders

- **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



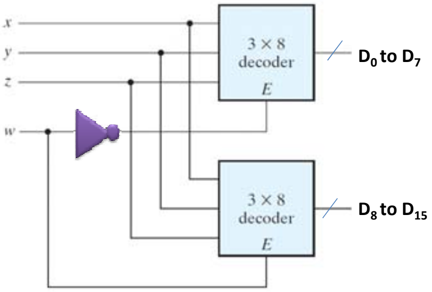
Decoders and Muxes

- 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



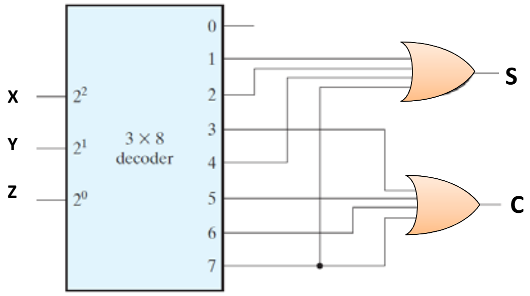
Boolean Function Implementation using Decoders

- **As Decoder covers all the Minterms**
- Using a n-to-2n decoder and OR gates any functions of n variables can be implemented.
- Example: Full Adder

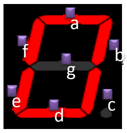
  S(x,y,z)= Σ(1,2,4,7) , C(x,y,z)=Σ(3,5,6,7)
- Functions S and C can be implemented using a 3-to-8 decoder and two 4-input OR gates

Decoder: Covers All Minterms

Implementation of S and C



Activation of LEDs

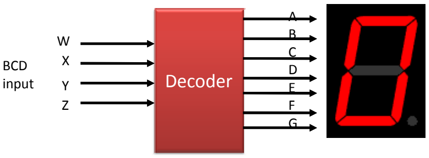


F_a(W,X,Y,Z)= Σm(0,2,3,5,6,7,8,9) + Σd(10, 11,12,13,14,15)

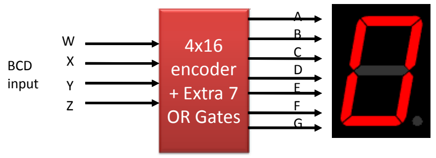
- 0 : a,b,c,d,e,f
- 1: b,c
- 2:a,b,g,e,d
- 3:a,b,g,c,d
- 4:f,g,b,c
- 5:a,f,g,c,d
- 6:a,f,g,c,d,e
- 7:a,b,c
- 8:a,b,c,d,e,f,g
- 9:a,b,c,d,f,g

BCD to 7 Segment Display

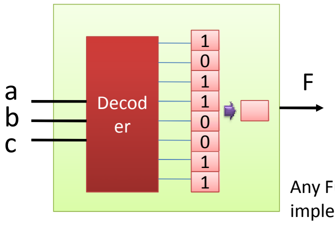
- BCD are 4 bit
- Design a decoder to drive 7 segment LED



BCD to 7 Segment Decoder using 4x16 Encoder



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

Encoders

- A digital circuit perform the inverse operation of Decoder
- 4x2 Encoder : 4 input lines A, B, C, D and two output lines X, Y

A	B	C	D	X	Y
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

X=A’B’CD’+A’B’C’D
=A’B’(CD’+C’D)

Y=A’BC’D’+A’B’C’D
=A’C’(BD’+B’D)

X=C+D
Y=B+D

Priority Encoders

- Encoder with priority function
- Two or more input is 1 at the same time: input with highest priority will take precedence
- When all the input are 0 it is invalid : V=0

A	B	C	D	X	Y	V
0	0	0	0	0	0	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1

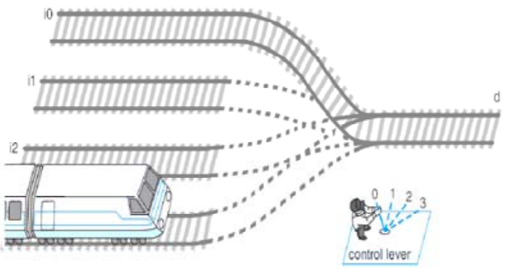
X=C+D
Y=D+BC’ **//mistake in class is => corrected…**
V=A+B+C+D

Multiplexor (Mux)

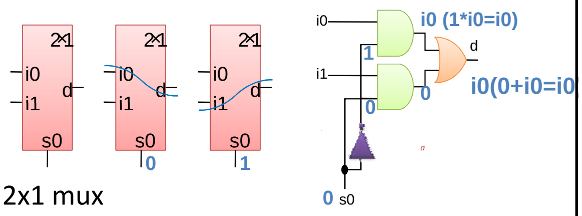
- Multiplex: Cinema, TDM, High BW
- Mux: Another popular combinational building block
  - Routes one of its N data inputs to its one output, based on binary value of select inputs
  - 4 input mux → needs 2 select inputs to indicate which input to route through
  - 8 input mux → 3 select inputs
  - **N inputs → $\log_2(N)$ selects**

Multiplexor (Mux)

- Mux: Another popular combinational building block
  - Like a railyard switch

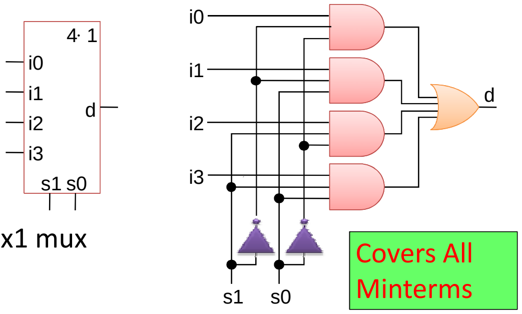


Mux Internal Design



2x1 mux

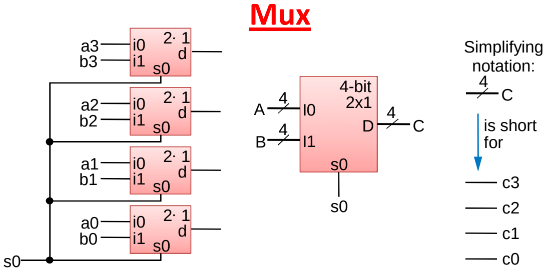
Mux Internal Design



4x1 mux

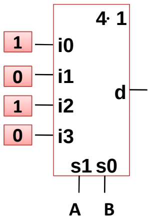
Covers All Minterms

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

Implementing logic Function using MUX



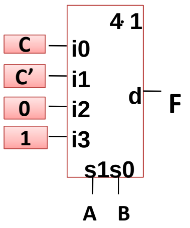
F (A, B) = ∑m(0,2)

4x1 mux

Implementing 3 Inputs Logic Function using 4x1 MUX

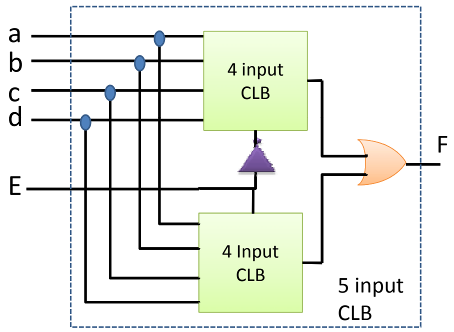
F (A, B,C) = ∑m(1,2,6,7)

A	B	C	F	F
0 0	0 0	0 1	0 1	C
0 0	1 1	0 1	1 0	C’
1 1	0 0	0 1	0 0	0
1 1	1 1	0 1	1 1	1



4x1 mux

Implementing 5 Inputs Logic Function using two 16x1 MUX


a
b
c
d
E
4 input CLB
4 Input CLB
F
5 input CLB