

Week 8 Discussion

5/20/2022

Isabella Liu

(Based on discussion slides from previous quarters)

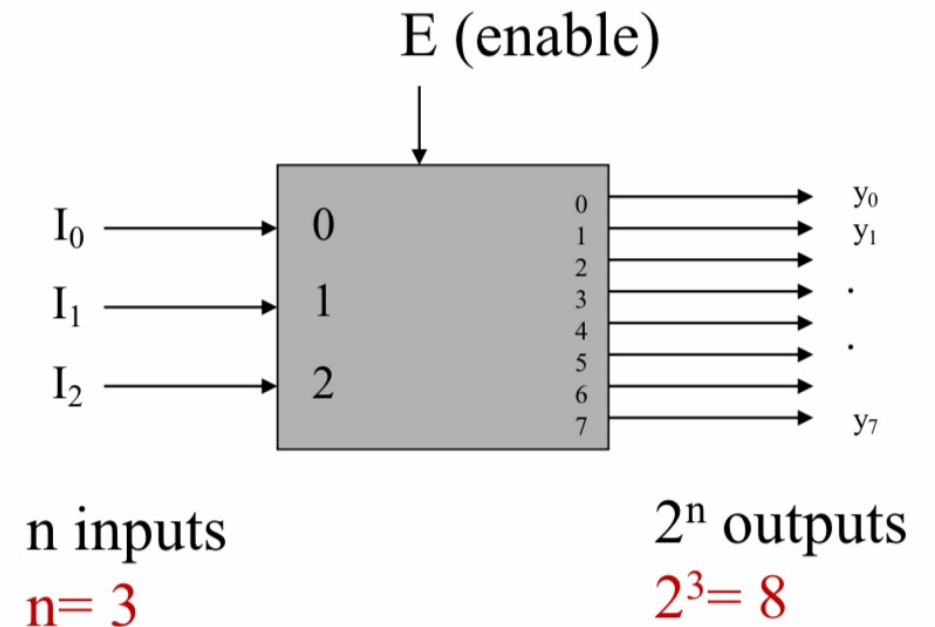
Overview

- Standard Combinational Modules
 - Decoder
 - Encoder
 - Multiplexer (Mux)
- System Designs

Decoder

- Definition

- A digital module that converts a binary address to the assertion of the addressed device
- $N \rightarrow 2^N$
- One-hot outputs
 - Only one output HIGH (1) at most
 - E.g. $N=2$, 0010, 0001, ... But not 0011, 0110, ...



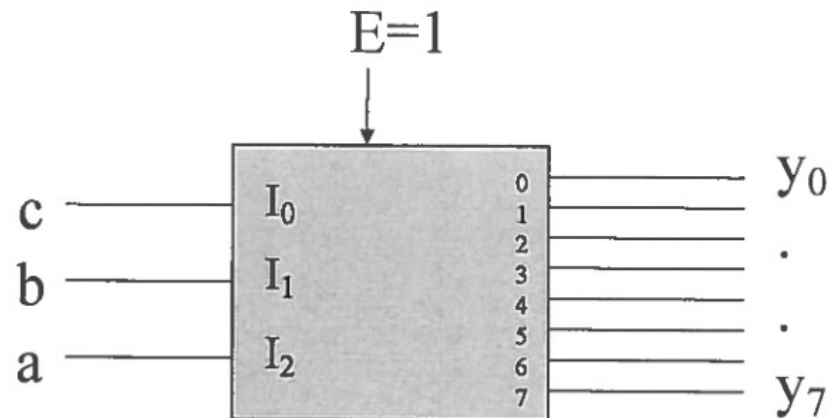
Decoder

- Use decoder to implement Boolean functions
 - Decoder produces minterms when $E=1$
 - Use OR gates to collect the minterms to cover the On-set (1)
 - For the Don't Care-Set (X), we can just ignore the terms

Decoder

- Use decode to implement Boolean functions
 - Example
 - Use a 3-input decoder and OR gates to implement

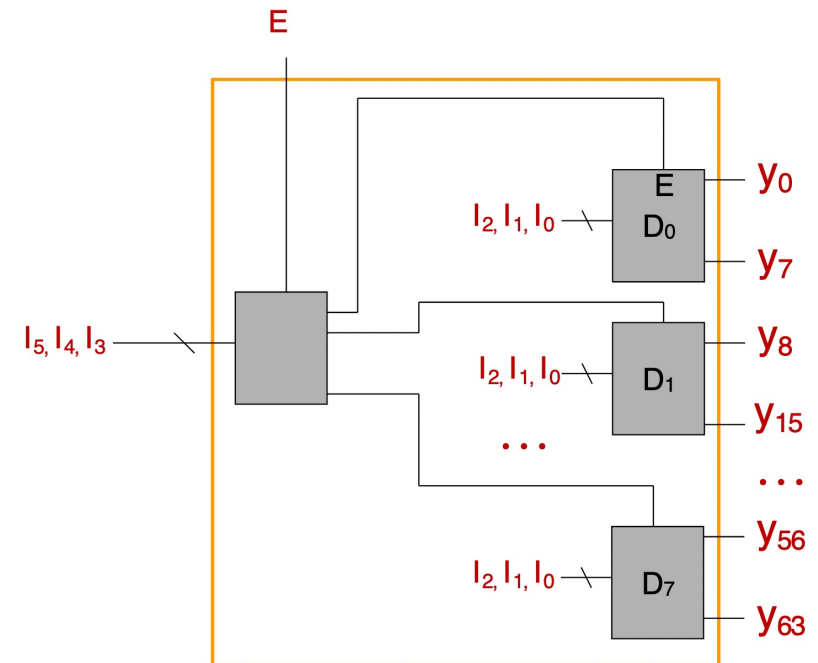
$$f(a, b, c) = \sum m(1, 2, 4) + \sum d(0, 5)$$



Decoder

- Tree structure

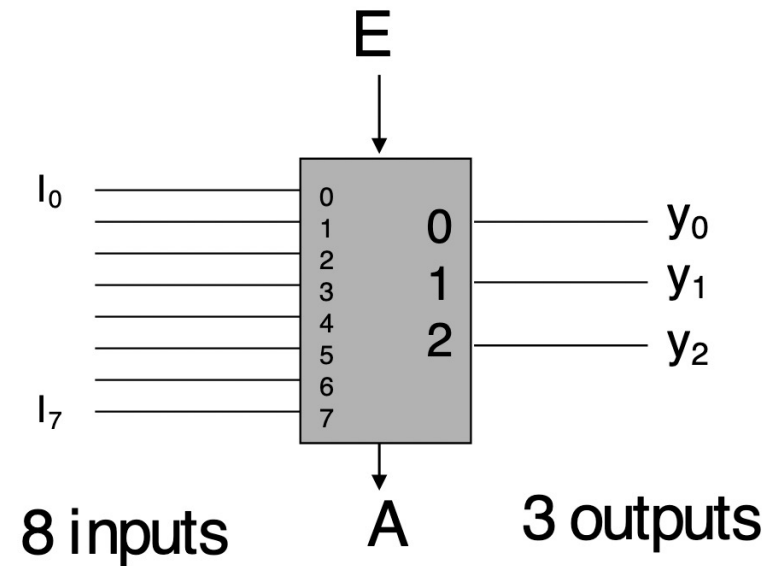
- Scale up the size of the decoders using a tree structure
- Example
 - Implement a $6 - 2^6$ decoder using $3 - 2^3$ decoders
 - # of expressing decoders: $2^6 - 2^3 = 8$
 - # of control bits: $\sqrt[2]{8} = 3$
 - # of control decoders: $\text{ceil}(3 / 3) = 1$



Encoder

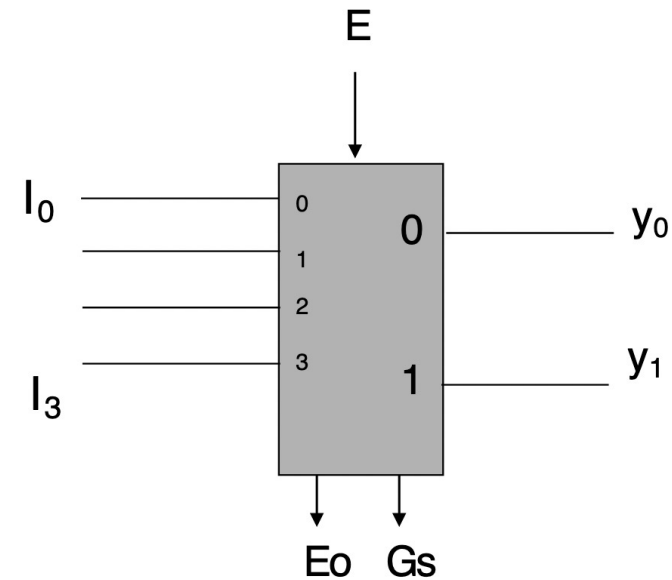
- Definition

- A digital module that converts the assertion of a device to the binary address of the device
- $2^N \rightarrow N$
- One-hot inputs



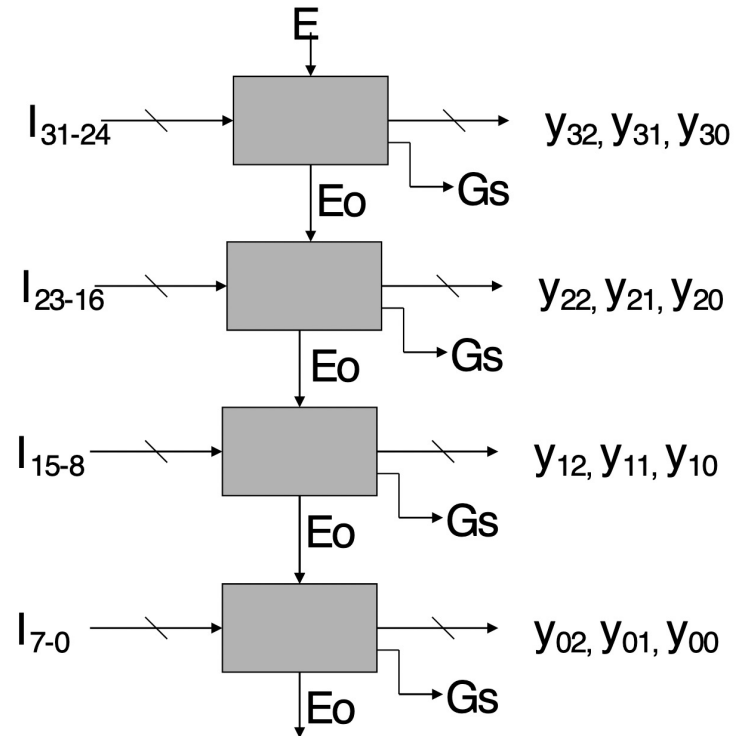
Priority Encoder

- Replace A with G_S
 - $G_S = 1$ if $E = 1$, and $\exists i$, s.t. $I_i = 1$
- Use E_O to pass on enable
 - $E_O = 1$ if $E = 1$, and $\forall i, I_i = 0$



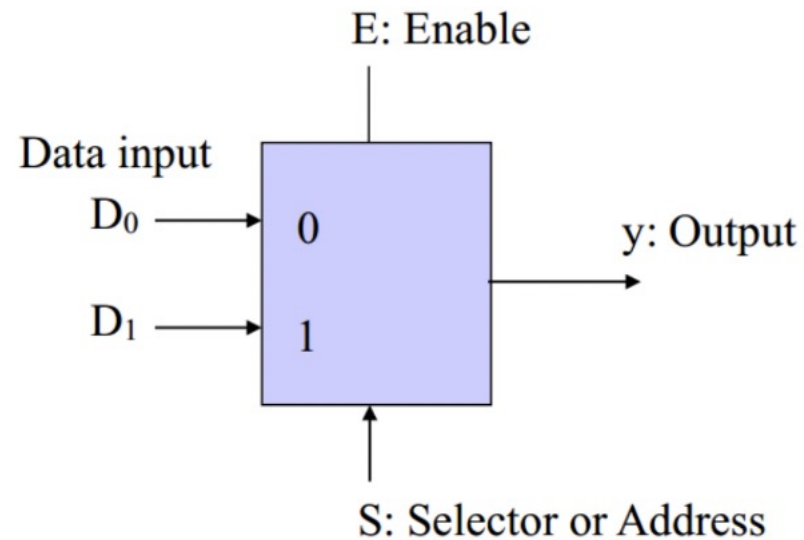
Priority Encoder

- Example
 - Implement a 32-input priority encoder with 8-input priority encoders (high bit priority)



Multiplexer (Mux)

- Definition
 - Selects one of N inputs to connect to the output
 - $\log_2(N)$ -bit select input – control input



Multiplexer (Mux)

- Use Mux to implement Boolean functions
 - Step 1: use selector to decompose the function into smaller functions (less number of variables) – Shannon's expansion
 - Step 2: Simplify the decomposed functions using K-map – Consensus theorem

Multiplexer (Mux)

- Use Mux to implement Boolean functions
 - Example
 - Implement the following Boolean function with 2:1 Mux

$$f(a, b, c) = \sum m(0, 3, 5, 7) + \sum d(6)$$

Multiplexer (Mux)

- Use Mux to implement Boolean functions

- Example

- Implement the following Boolean function with 2:1 Mux

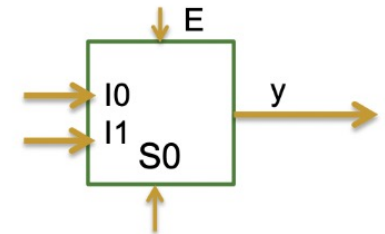
$$f(a, b, c) = \sum m(0, 3, 5, 7) + \sum d(6)$$

- Step 1: use selector to decompose the function into smaller functions (less number of variables) – Shannon's expansion

- For $N:\log_2(N)$ Mux, decompose $\log_2(N)$ -bit input each time

id	a	b	c	f
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	x
7	1	1	1	1

bc	00	01	10	11	
a=0	1	0	0	1	$I0 = d0(bc)$
a=1	0	1	x	1	$I1 = d1(bc)$



Multiplexer (Mux)

- Use Mux to implement Boolean functions

- Example

- Implement the following Boolean function with 2:1 Mux

$$f(a, b, c) = \sum m(0, 3, 5, 7) + \sum d(6)$$

- Step 1: use selector to decompose the function into smaller functions (less number of variables) – Shannon's expansion

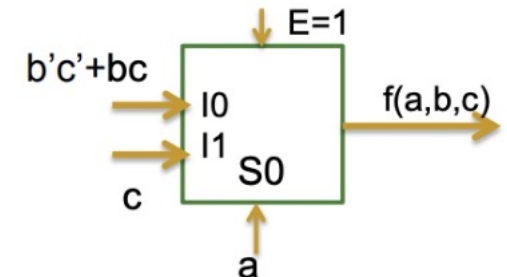
- For $N:\log_2(N)$ Mux, decompose $\log_2(N)$ -bit input each time

id	a	b	c	f
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	x
7	1	1	1	1

bc	00	01	10	11	
a=0	1	0	0	1	$I_0 = d_0(bc)$
a=1	0	1	x	1	$I_1 = d_1(bc)$

c\b	0	1
0	1	0
1	0	1

c\b	0	1
0	0	x
1	1	1



Multiplexer (Mux)

- Use Mux to implement Boolean functions

- Example

- Implement the following Boolean function with 2:1 Mux

$$f(a, b, c) = \sum m(0, 3, 5, 7) + \sum d(6)$$

- Step 1: use selector to decompose the function into smaller functions (less number of variables) – Shannon's expansion

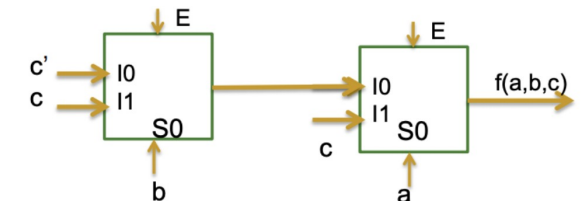
- For $N:\log_2(N)$ Mux, decompose $\log_2(N)$ -bit input each time

id	a	b	c	f
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	x
7	1	1	1	1

bc	00	01	10	11	
a=0	1	0	0	1	$I0 = d0(bc)$
a=1	0	1	x	1	$I1 = d1(bc)$

c\b	0	1	c\b	0	1
0	1	0	0	0	x
1	0	1	1	1	1

c	0	1	
b=0	1	0	$I0 = c'$
b=1	0	1	$I1 = c$



Thanks