NEC 304

STLD

Lecture 17

Encoders and Decoders

Rajeev Pandey

Department Of ECE

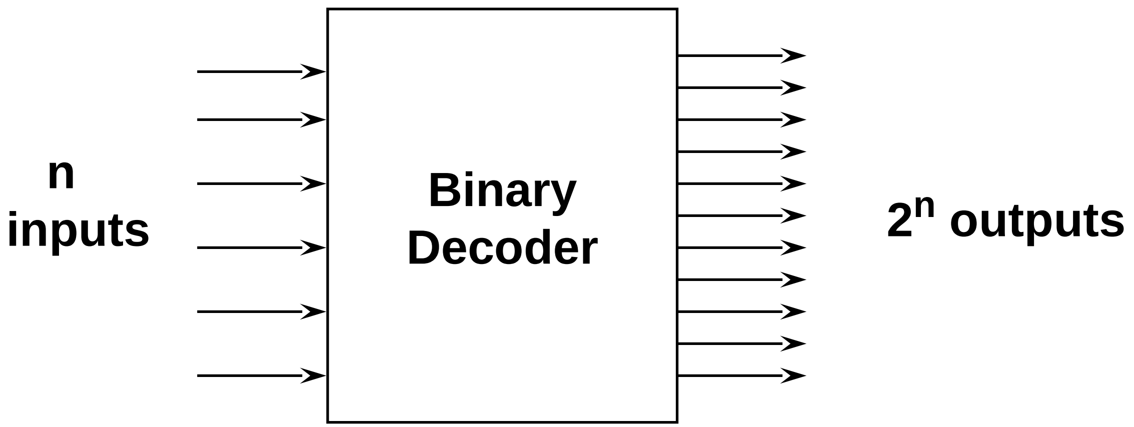
rajeevvce2007@gmail.com

Overview

- **Binary decoders**
  - **Converts an n-bit code to a single active output**
  - **Can be developed using AND/OR gates**
  - **Can be used to implement logic circuits.**
- **Binary encoders**
  - **Converts one of 2^n inputs to an n-bit output**
  - **Useful for compressing data**
  - **Can be developed using AND/OR gates**
- **Both encoders and decoders are extensively used in digital systems**

Binary Decoder

- Black box with n input lines and 2^n output lines
- Only one output is a 1 for any given input

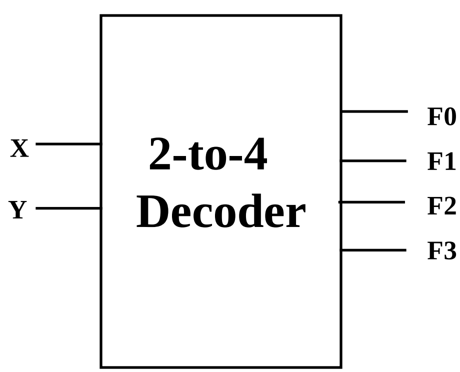


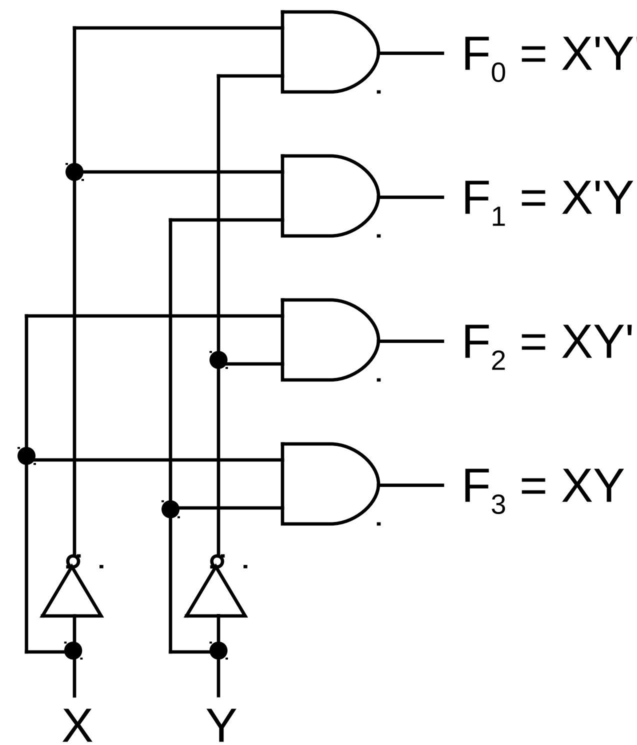
2-to-4 Binary Decoder

Truth Table:

X	Y	F_0	F_1	F_2	F_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

- **From truth table, circuit for 2x4 decoder is:**
- **Note: Each output is a 2-variable minterm (**X'Y', X'Y, XY' **or** XY**)**

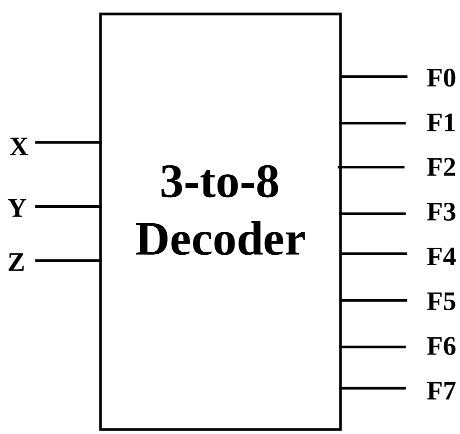


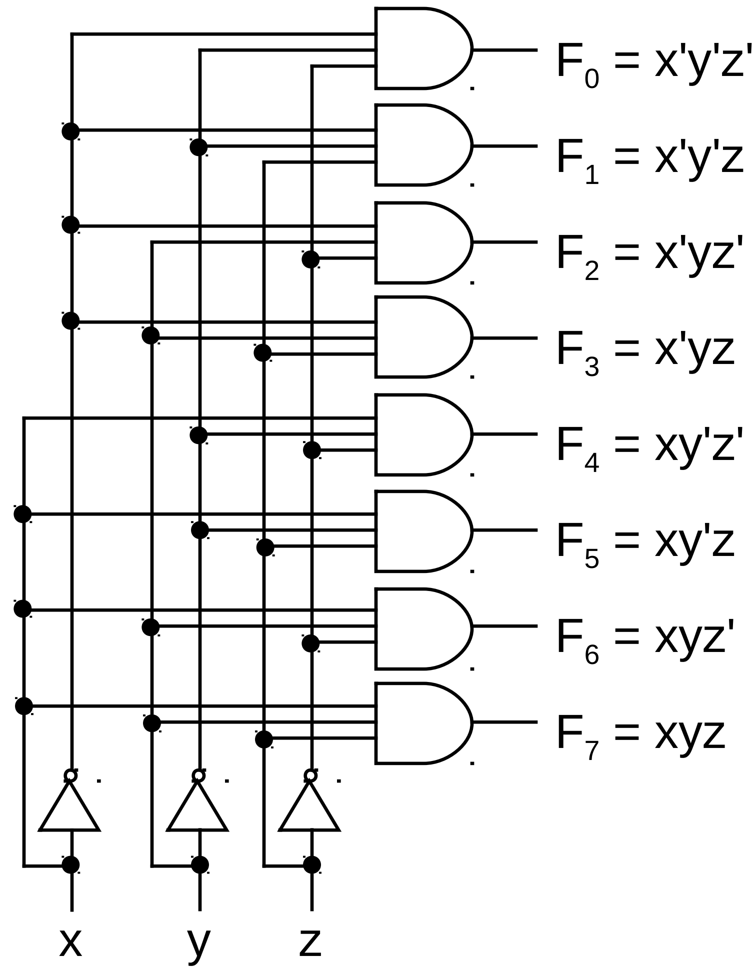


3-to-8 Binary Decoder

Truth Table:

x	y	z	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1





Implementing Functions Using Decoders

- **Any n-variable logic function can be implemented using a single n-to-2^n decoder to generate the minterms**
  - **OR gate forms the sum.**
  - **The output lines of the decoder corresponding to the minterms of the function are used as inputs to the or gate.**
- **Any combinational circuit with *n* inputs and *m* outputs can be implemented with an n-to-2^n decoder with *m* OR gates.**
- **Suitable when a circuit has many outputs, and each output function is expressed with few minterms.**

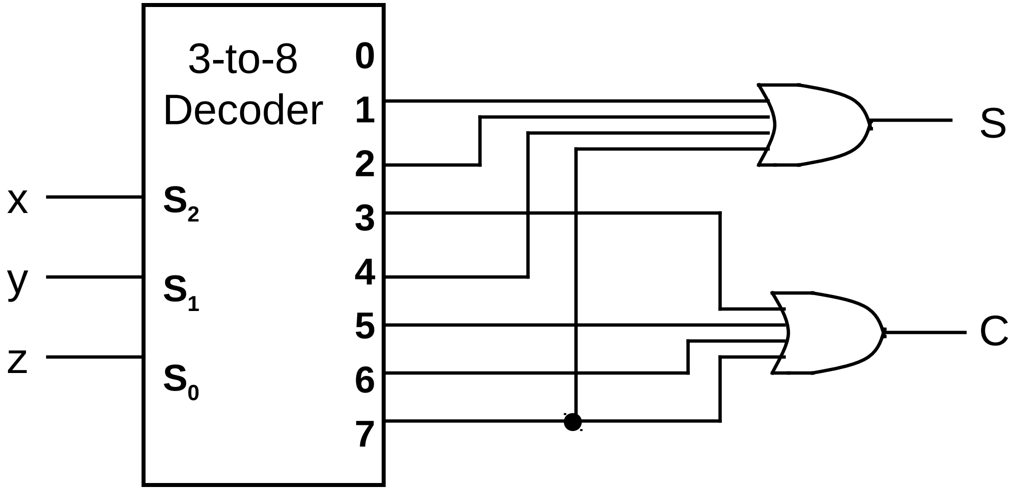
Implementing Functions Using Decoders

- **Example: Full adder**

  S(x, y, z) = Σ (1,2,4,7)

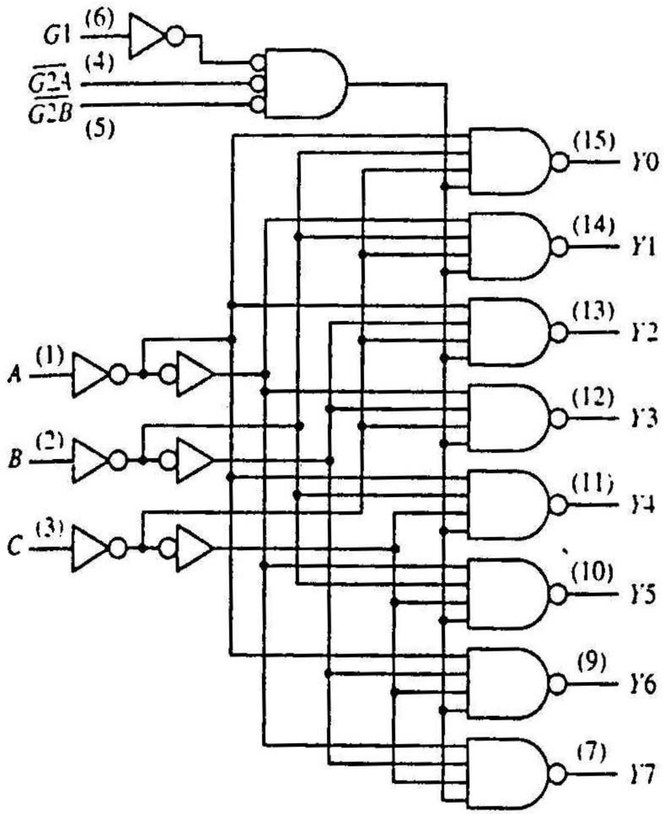
  C(x, y, z) = Σ (3,5,6,7)

x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

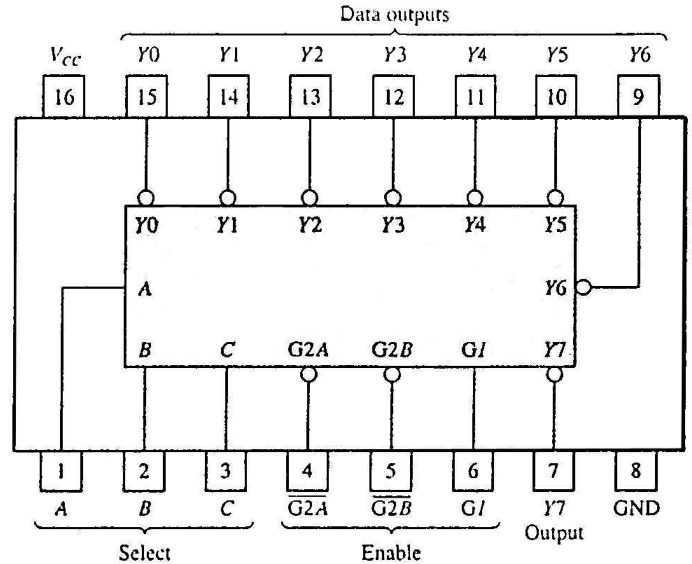


Standard MSI Binary Decoders Example

74138 (3-to-8 decoder)



(a)



(b)

Inputs					Outputs							
Enable		Select										
G1	$\overline{G2}$*	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L
×	H	×	×	×	H	H	H	H	H	H	H	H
L	×	×	×	×	H	H	H	H	H	H	H	H

$\overline{G2}^* = \overline{G2A} + \overline{G2B}$

(c)

(a) Logic circuit.
(b) Package pin configuration.
(c) Function table.

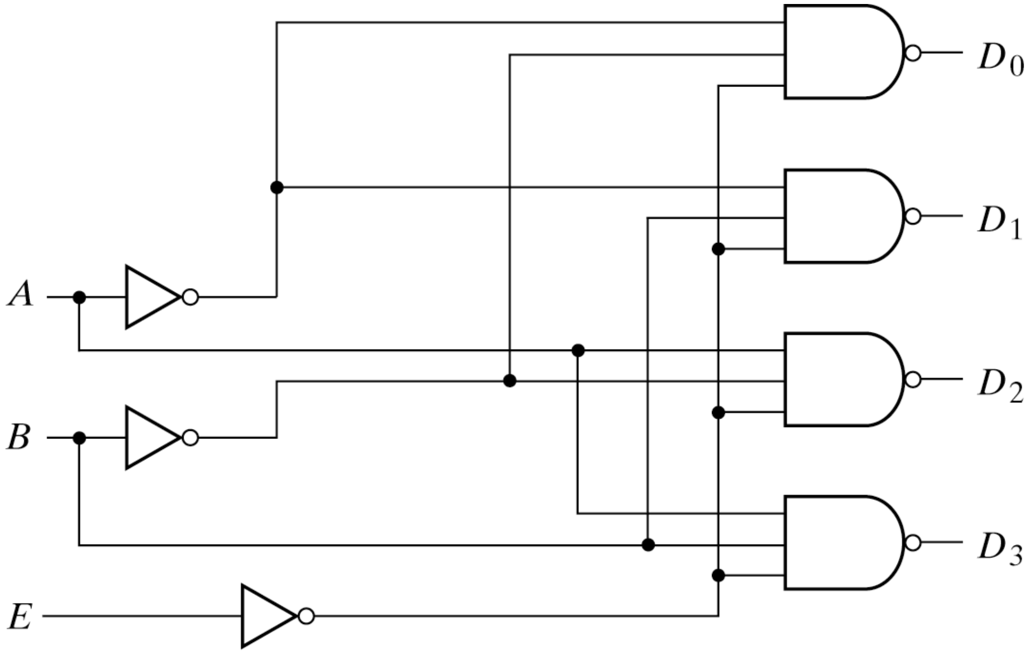
Building a Binary Decoder with NAND Gates

- Start with a 2-bit decoder
  - Add an enable signal (E)

Note: use of NANDs

only one 0 active!

if E = 0



(a) Logic diagram

E	A	B	D_0	D_1	D_2	D_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

(b) Truth table

Fig. 4-19 2-to-4-Line Decoder with Enable Input

Use two 3 to 8 decoders to make 4 to 16 decoder

- Enable can also be active high
- In this example, only one decoder can be active at a time.
- x, y, z effectively select output line for w

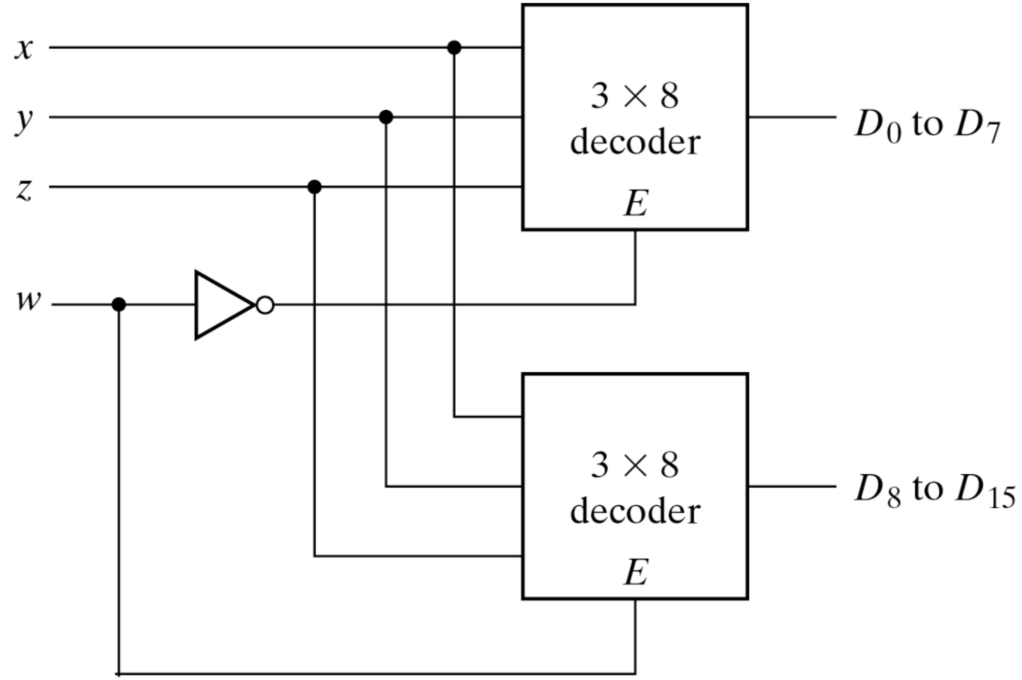


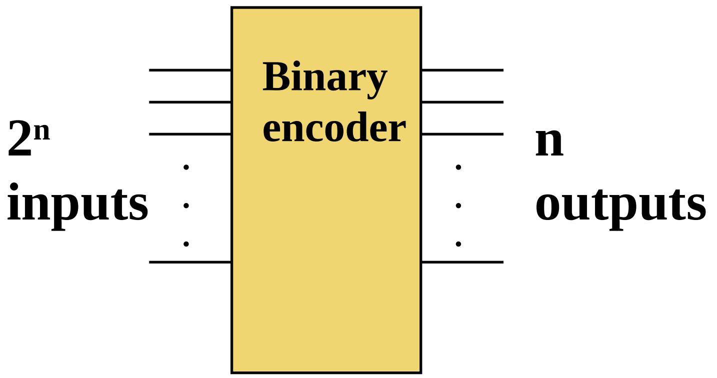
Fig. 4-20 4 × 16 Decoder Constructed with Two 3 × 8 Decoders

Encoders

- **If the a decoder's output code has fewer bits than the input code, the device is usually called an encoder.**

  e.g. 2^n-to-n

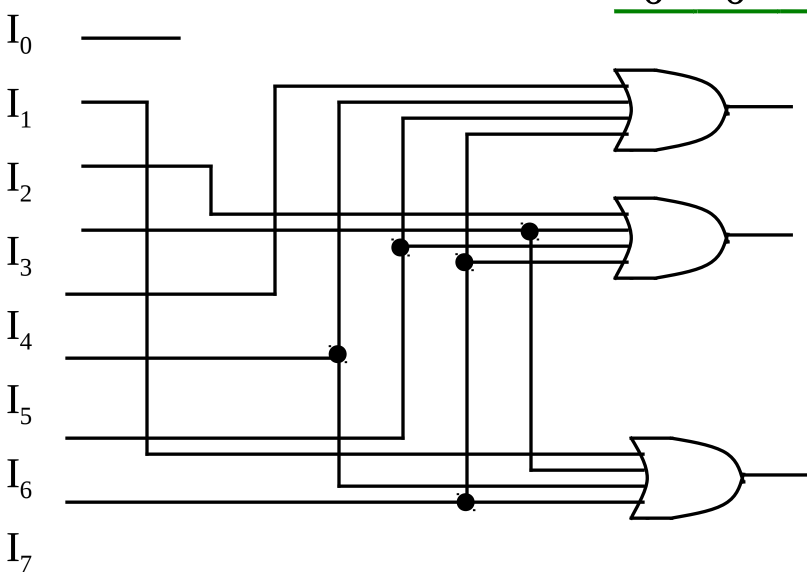
- **The simplest encoder is a 2^n-to-n binary encoder**
  - **One of 2^n inputs = 1**
  - **Output is an n-bit binary number**



8-to-3 Binary Encoder

At any one time, only one input line has a value of 1.

Inputs								Outputs		
I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	y_2	y_1	y_0
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1



$y_2 = I_4 + I_5 + I_6 + I_7$

$y_1 = I_2 + I_3 + I_6 + I_7$

$y_0 = I_1 + I_3 + I_5 + I_7$

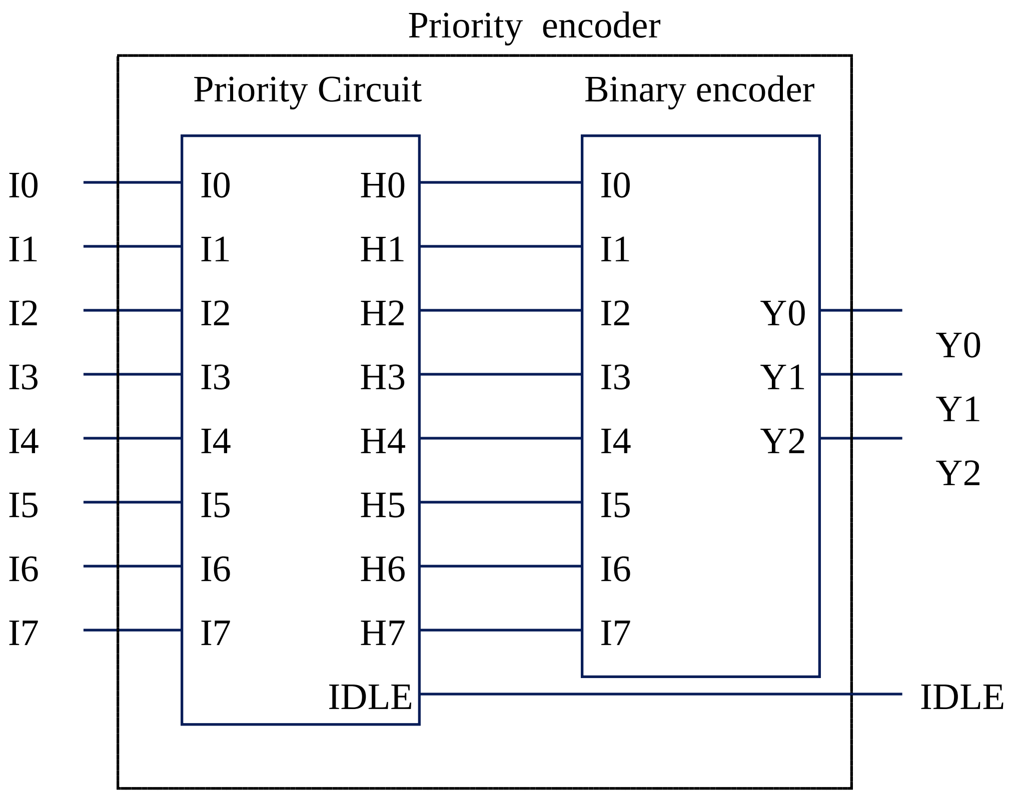
8-to-3 Priority Encoder

- **What if more than one input line has a value of 1?**
- **Ignore “lower priority” inputs.**
- **Idle indicates that no input is a 1.**
- **Note that polarity of Idle is opposite from Table 4-8 in Mano**

Inputs								Outputs			
I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	y_2	y_1	y_0	Idle
0	0	0	0	0	0	0	0	x	x	x	1
1	0	0	0	0	0	0	0	0	0	0	0
X	1	0	0	0	0	0	0	0	0	1	0
X	X	1	0	0	0	0	0	0	1	0	0
X	X	X	1	0	0	0	0	0	1	1	0
X	X	X	X	1	0	0	0	1	0	0	0
X	X	X	X	X	1	0	0	1	0	1	0
X	X	X	X	X	X	1	0	1	1	0	0
X	X	X	X	X	X	X	1	1	1	1	0

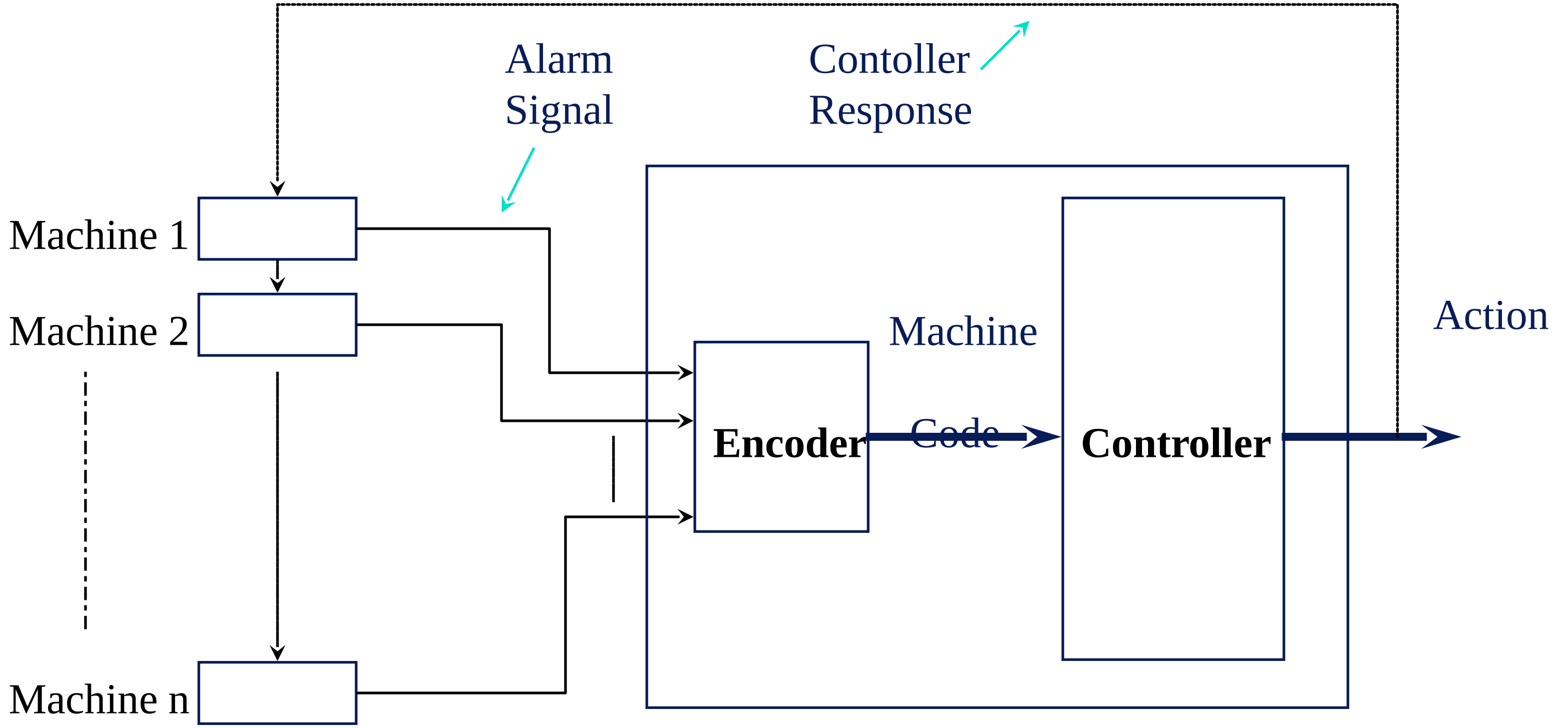
Priority Encoder (8 to 3 encoder)

- **Assign priorities to the inputs**
- **When more than one input are asserted, the output generates the code of the input with the highest priority**
- **Priority Encoder :**
  - H7=I7 (Highest Priority)
  - H6=I6.I7’
  - H5=I5.I6’.I7’
  - H4=I4.I5’.I6’.I7’
  - H3=I3.I4’.I5’.I6’.I7’
  - H2=I2.I3’.I4’.I5’.I6’.I7’
  - H1=I1. I2’.I3’.I4’.I5’.I6’.I7’
  - H0=I0.I1’. I2’.I3’.I4’.I5’.I6’.I7’
  - IDLE= I0’.I1’. I2’.I3’.I4’.I5’.I6’.I7’
- **Encoder**
  - Y0 = I1 + I3 + I5 + I7
  - Y1 = I2 + I3 + I6 + I7
  - Y2 = I4 + I5 + I6 + I7



Encoder Application (Monitoring Unit)

- **Encoder identifies the requester and encodes the value**
- **Controller accepts digital inputs.**



Summary

- Decoder allows for generation of a single binary output from an input binary code
  - For an n-input binary decoder there are 2^n outputs
- Decoders are widely used in storage devices (e.g. memories)
  - We will discuss these in a few weeks
- Encoders all for data compression
- Priority encoders rank inputs and encode the highest priority input
- Next time: storage elements!