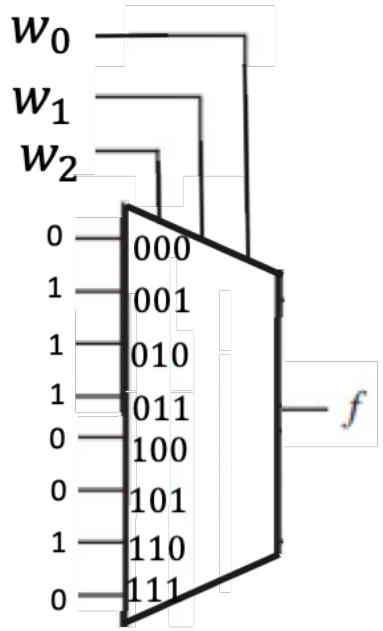
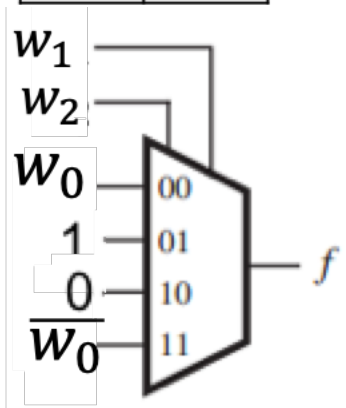
Basic Functions Using Multiplexers:

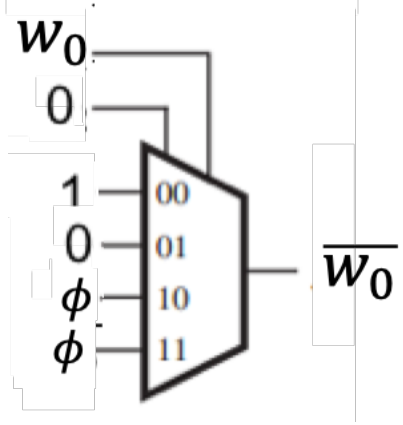
$$f(w_2, w_1, w_0) = \overline{w_2} \cdot w_0 + w_1 \cdot \overline{w_0}$$

w_2 \ w_1w_0	00	01	11	10
0	0	1	1	1
1	0	0	0	1

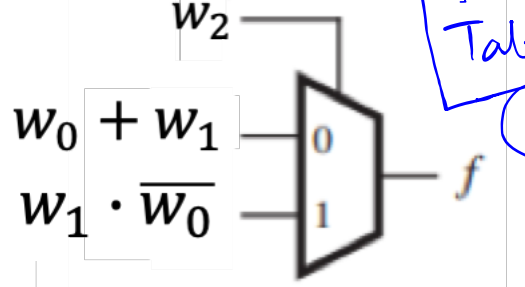


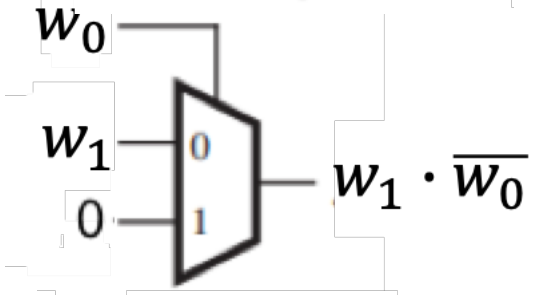
w_2 \ w_1	0	1
0	w_0	1
1	0	$\overline{w_0}$

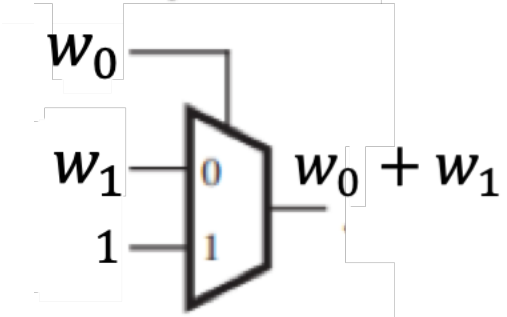




w_2	
0	$w_0 + w_1$
1	$w_1 \cdot \overline{w_0}$





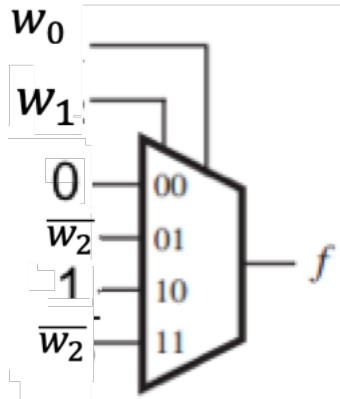


from Table

draw K-map

Alternative:

w_1w_0	00	01	11	10
0	$\overline{w_2}$	$\overline{w_2}$	1	



w_0	0	1
	w_1	$\overline{w_2}$

Boolean Functions Using Multiplexers:

- $f(w,x,y,z) = \sum m\,(1,4,5,7,9,10,12,13)$ using 4:1 MUX and 2:1 MUX only

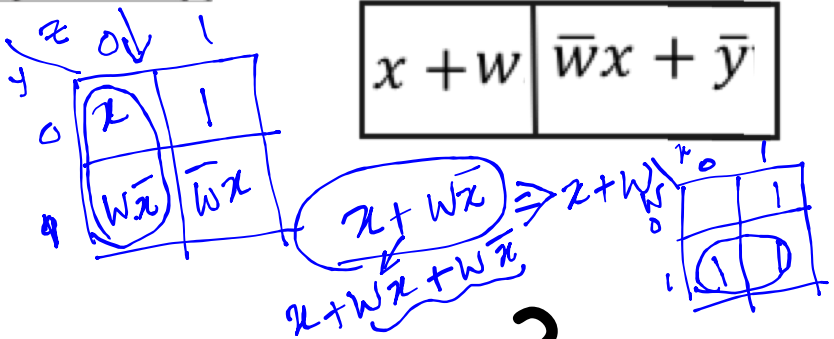
wx \ yz	00	01	11	10
	m_0	m_1	m_3	m_2
00	0	1	0	0
	m_4	m_5	m_7	m_6
01	1	1	1	0
	m_{12}	m_{13}	m_{15}	m_{14}
11	1	1	0	0
	m_8	m_9	m_{11}	m_{10}
10	0	1	0	1

yz	00	01	11	10
	m_0	m_1	m_3	m_2
	x	1	$\bar{w}x$	$w\bar{x}$

wx	yz
00	$\bar{y}z$
01	$\bar{y}+z$
11	$\bar{y}$
10	$\bar{y}z+y\bar{z}$

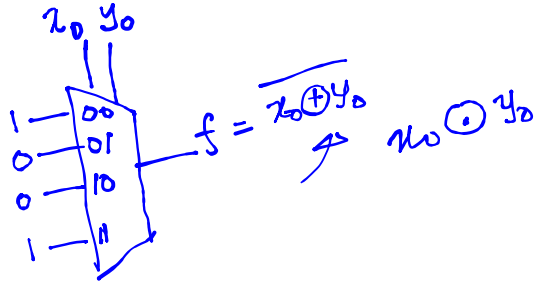
$y=0$	$y=1$
$x+z$	$z\bar{w}x+\bar{z}w\bar{x}$

$z=0$	$z=1$
$x+w$	$\bar{w}x+\bar{y}$



Boolean Circuits Using Multiplexers (HW)

- $f(w,x,y,z) = \sum m\,(1,4,5,7,9,12,13)$ using 8:1 MUX and logic gates only
- Design 1-bit full adder using 4:1 multiplexers
- Design a 2-input XNOR gate using 4:1 multiplexer
- Design a 2-input XNOR gate using 2:1 multiplexers



3:8 Decoder with Enable Using 2:4 Decoder:

← 2 of these

control $\bar{w}_2$ $E_{n2} = E_n \bar{w}_2$ y_2 $E_{n2} = E_n w_2$ for a combn of w's corresponding $y \to 1$ ↓ sub. is dec. equivalent of $w_1 w_0$

w_1 w_0 → y_0 y_1 y_2 y_3

00 → 0 → y_0

10 → 2 → y_2

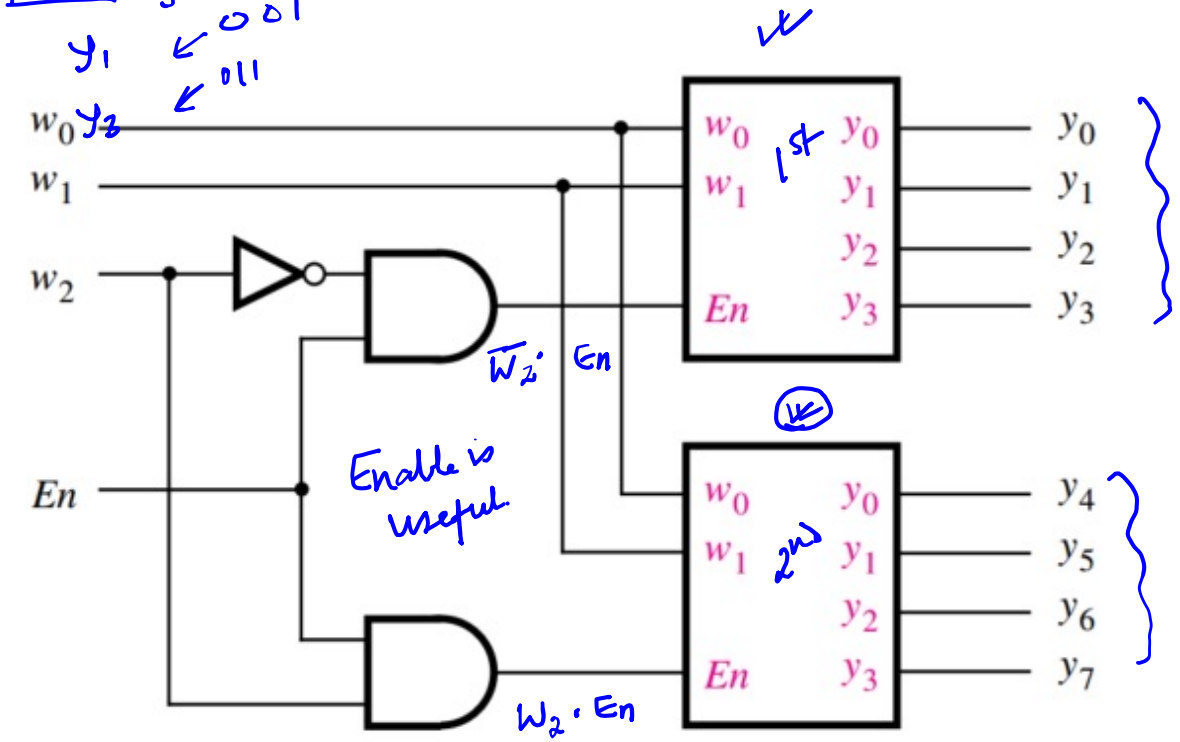
$w_1 w_2 w_3$

001

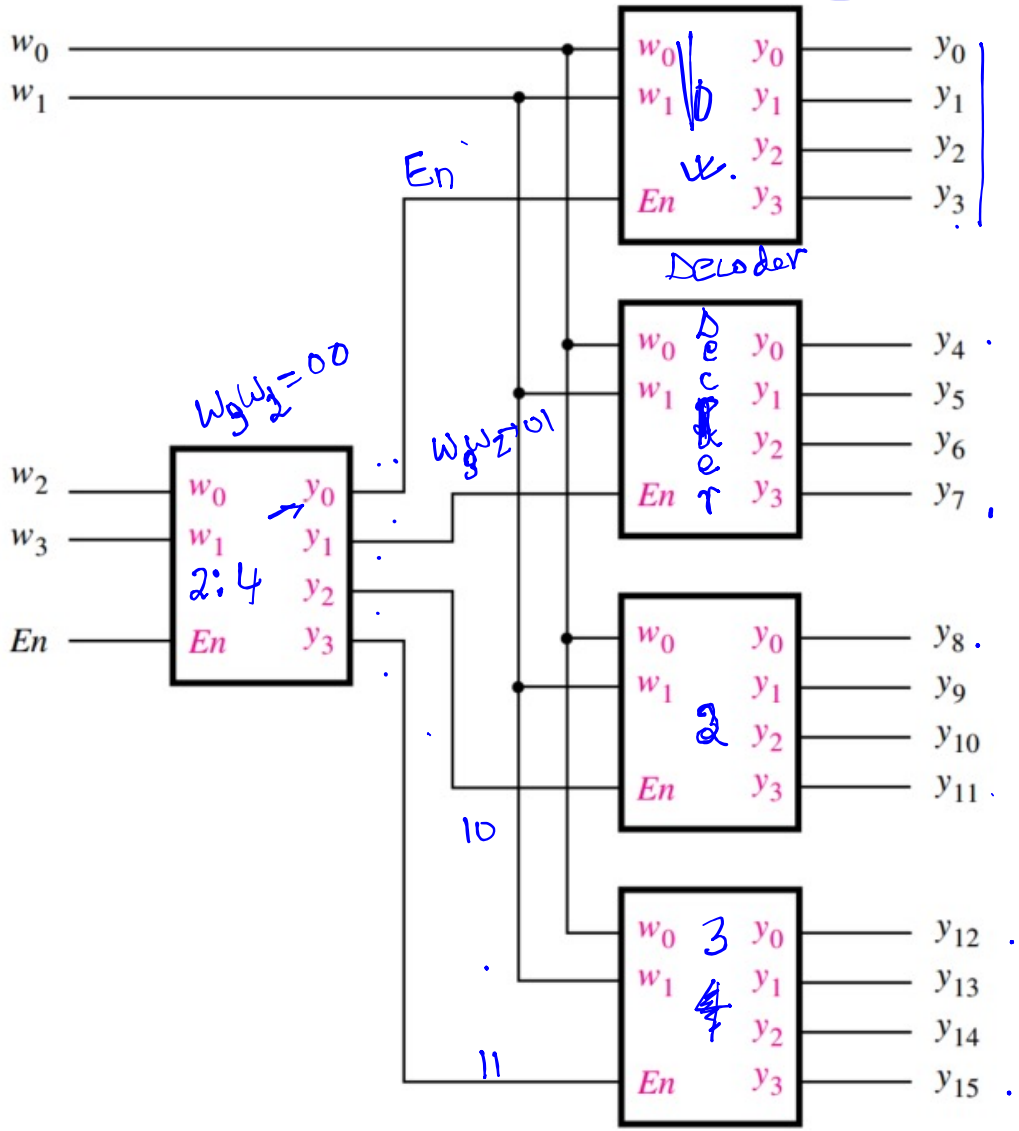
011

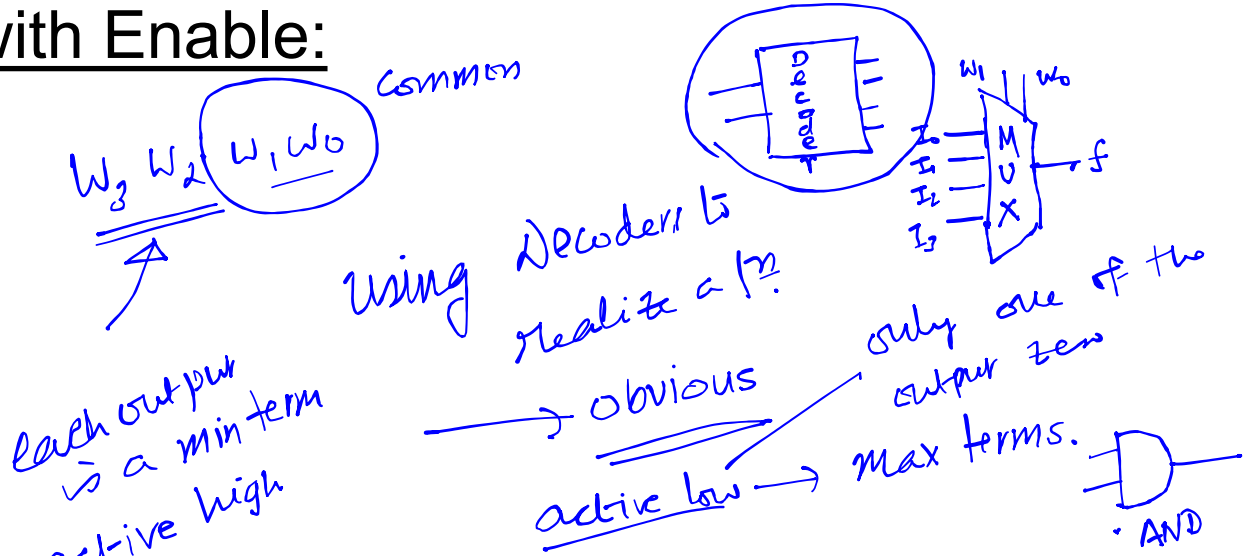
y_1

E_n	w_2	w_1	w_0	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7
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	0	0	0	1	0	0	0	0	0
1	0	1	1	0	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	1	0	0	0
1	1	0	1	0	0	0	0	0	1	0	0
1	1	1	0	0	0	0	0	0	0	1	0
1	1	1	1	0	0	0	0	0	0	0	1
0	ϕ	ϕ	ϕ	0	0	0	0	0	0	0	0



4:16 Decoder using 2:4 Decoders with Enable:

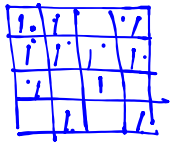




Homework: 4:16 Decoder using 3:8 Decoder with Enable.

Points to note while Realizing Logic Functions with Decoders:

- A function, f, with a long list of minterms requires an OR gate with large number of inputs. A n-variable function with k minterms can be expressed in its complemented form with $(2^n - k)$ minterms.

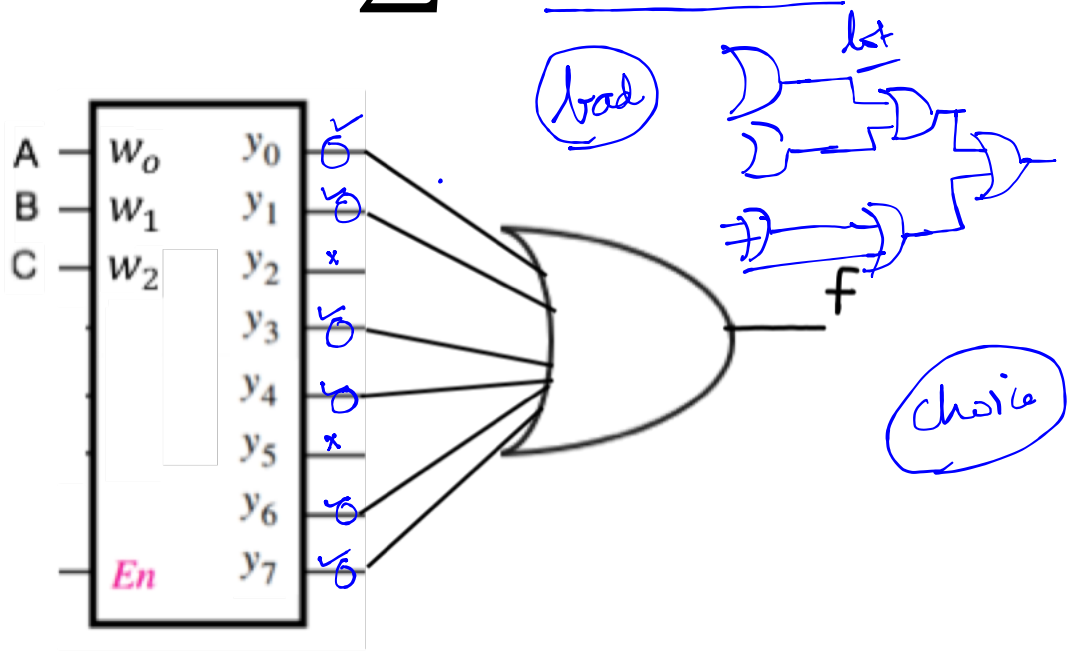


- If the number of minterms in a function with n-variables is greater than $2^n/2$ terms, then $\bar{f}$ can be expressed with fewer minterms. In such a case it is more advantageous to use NOR gate to sum the minterms of $\bar{f}$. The output of the NOR gate will be f.

- If the Decoder was made with NAND gates with Active Low Enable and Active Low Output, the external gates used must be NAND gates to get the SOP form instead of the OR gate. A two-level NAND gate circuit implements the SOP function and is equivalent to a two-level AND-OR circuit.

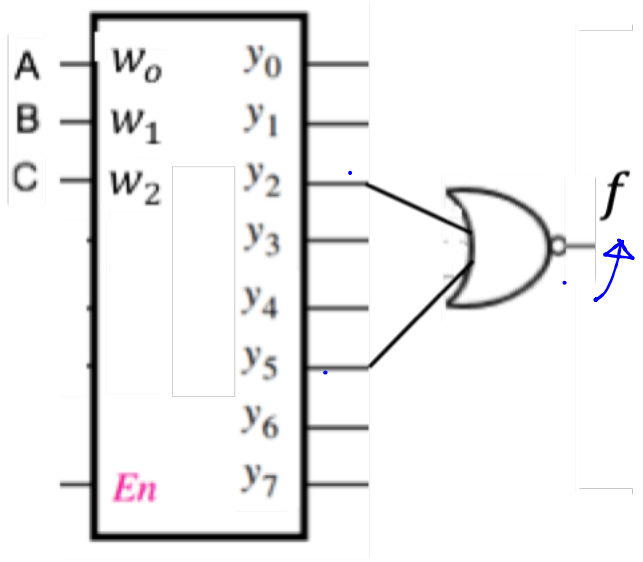
Logic Realization with Decoders:

$$f(A,B,C) = \sum m\,(0,1,3,4,6,7)$$



Choosing minterms for f

$$\bar{f}(A,B,C) = \sum m(2,5)$$



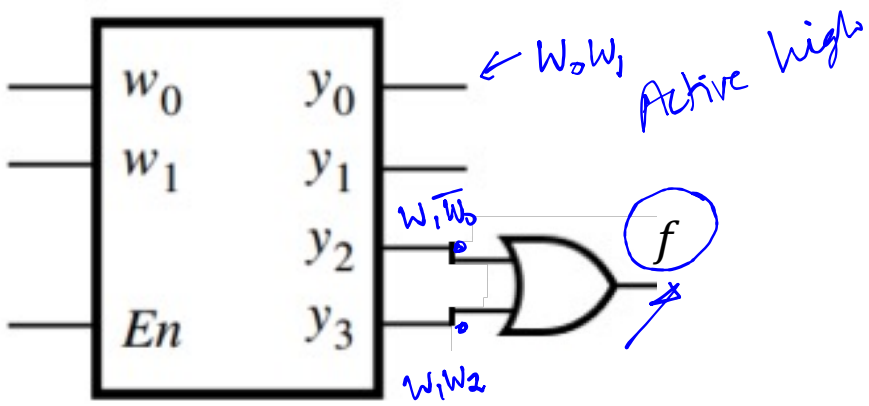
Choosing minterms for $\bar{f}$

Logic Realization with Decoders:

- $f(A, B) = \sum m(2,3)$

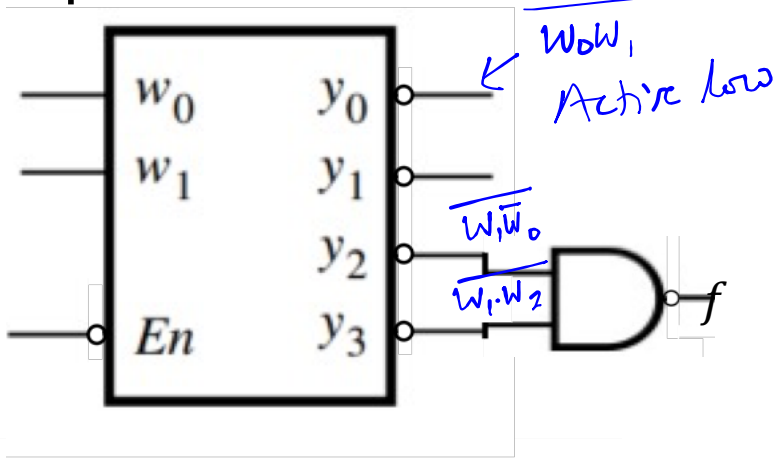
2 different ways.

The Decoder is always Enabled when output is to be used.



En	w_1	w_0	y_0	y_1	y_2	y_3
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1
0	ϕ	ϕ	0	0	0	0

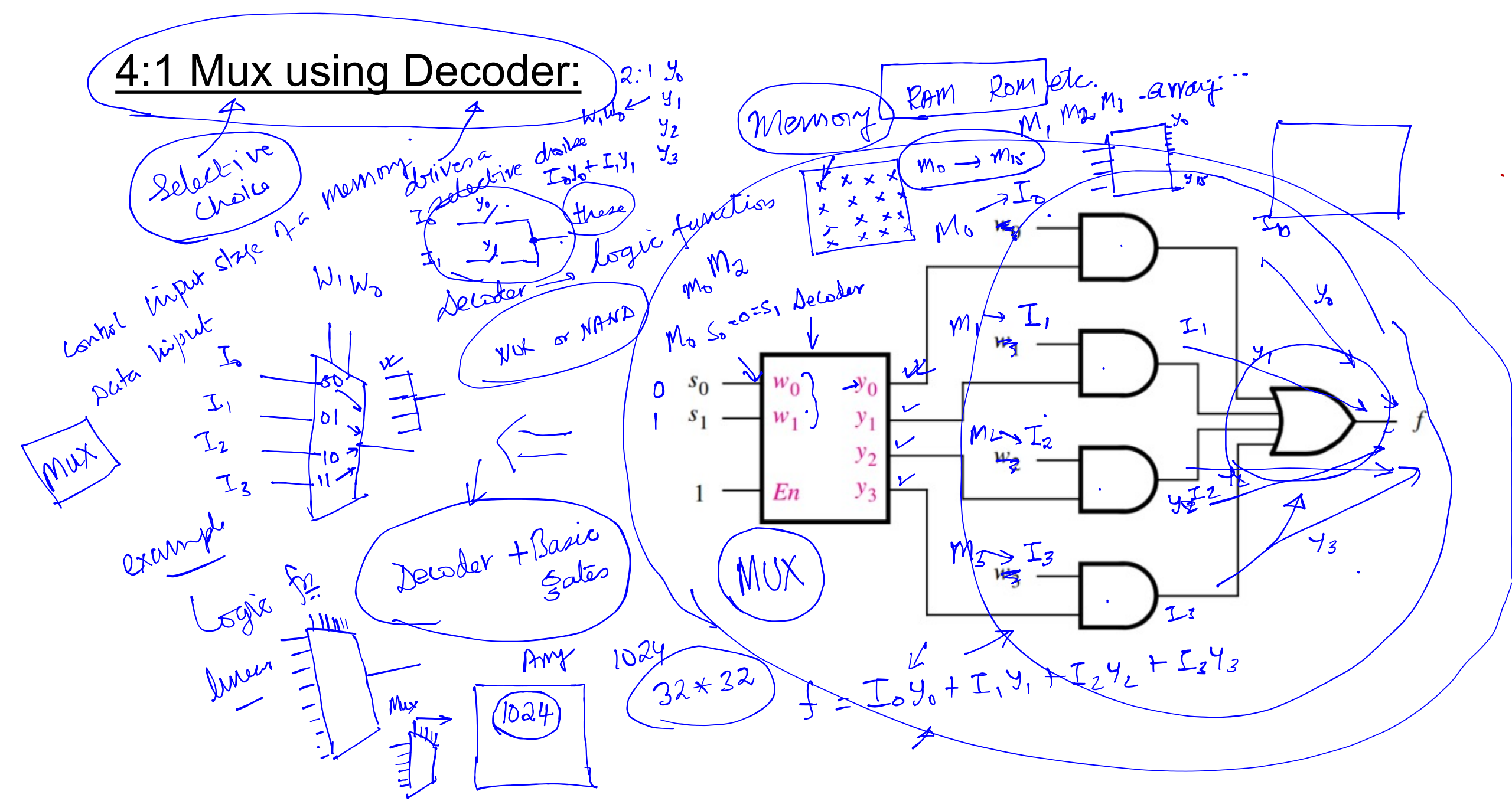
Active High Output.



En	w_1	w_0	y_0	y_1	y_2	y_3
1	ϕ	ϕ	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

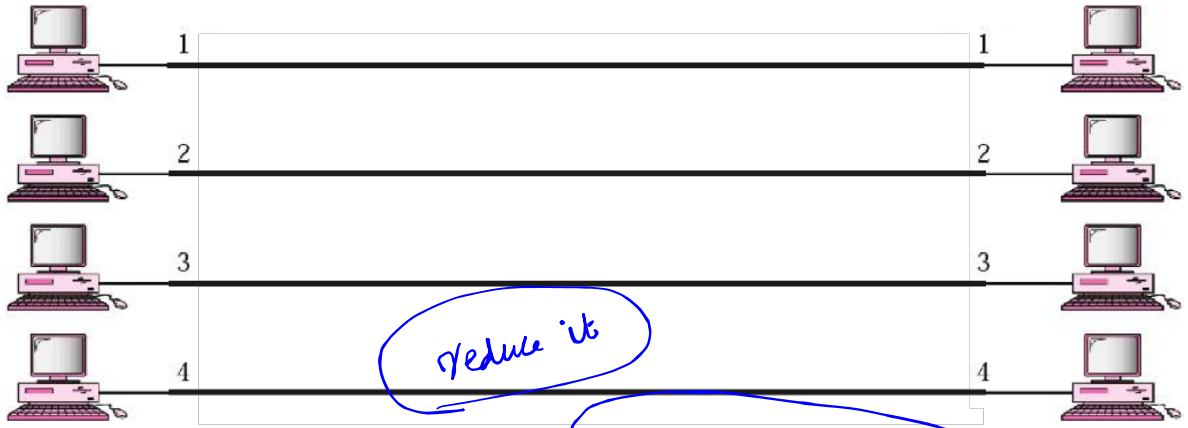
Active Low Output.

4:1 Mux using Decoder:

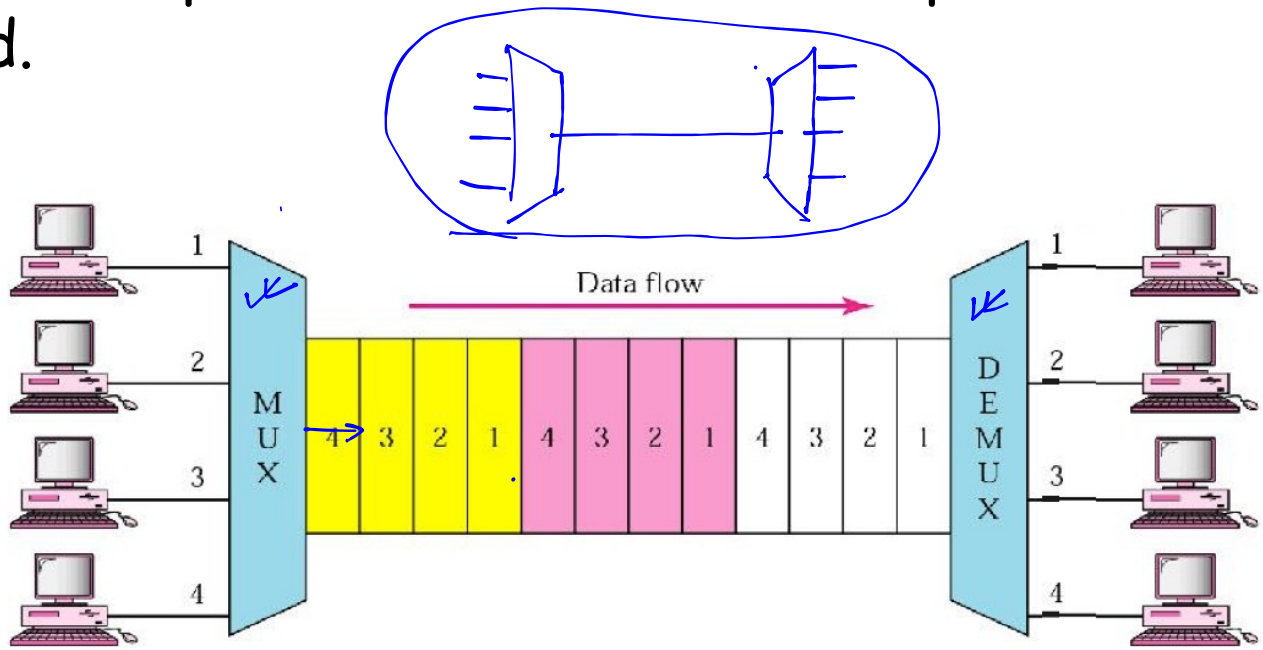


Demultiplexers:

A **demultiplexer** (also known as a **demux** or **data distributor**) is defined as a circuit that can distribute or deliver multiple outputs from a single input. The demultiplexer's output lines are 'n' in number, the select line number is 'm' and n = 2^m. The control signal or select input code decides the output line to which the input is to be transmitted.

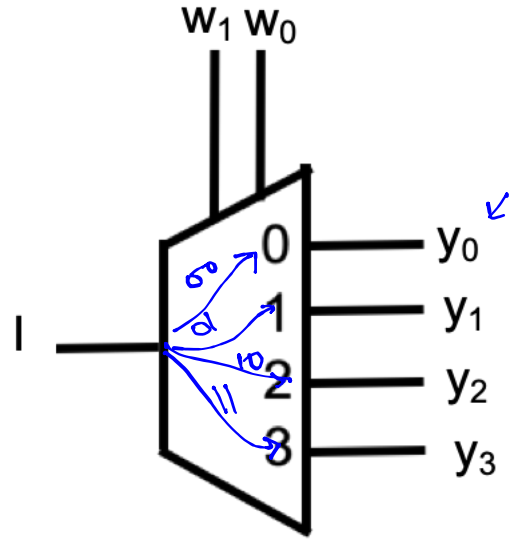


This requires 4 wires/lines



This requires 1 MUX, 1 wire and 1 DMUX

Demultiplexers:



Truth Table

w_1	w_0	y_0	y_1	y_2	y_3
0	0	I	0	0	0
0	1	0	I	0	0
1	0	0	0	I	0
1	1	0	0	0	I

$$y_0 = \overline{w_1} \cdot \overline{w_0} \cdot I$$

$$y_1 = \overline{w_1} \cdot w_0 \cdot I$$

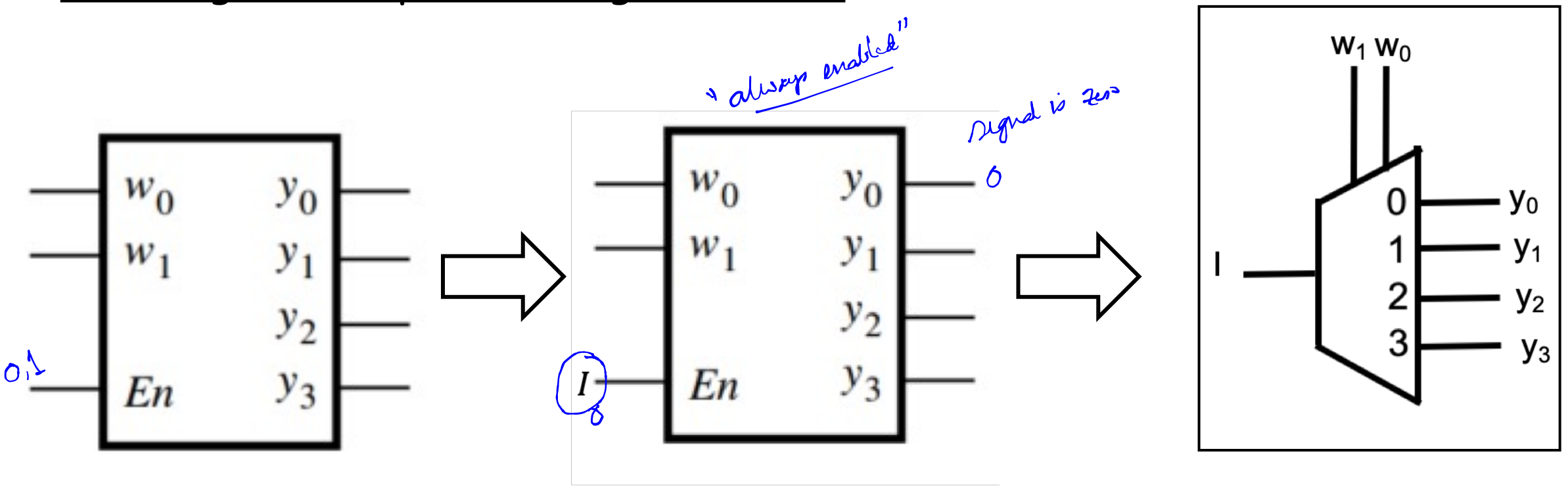
$$y_2 = w_1 \cdot \overline{w_0} \cdot I$$

$$y_3 = w_1 \cdot w_0 \cdot I$$

Decoder with Enable.

$I = E_n$

Realizing Demultiplexer using a Decoder:



$$y_0 = \overline{w_1} \cdot \overline{w_0} \cdot E_n$$
$$y_1 = \overline{w_1} \cdot w_0 \cdot E_n$$
$$y_2 = w_1 \cdot \overline{w_0} \cdot E_n$$
$$y_3 = w_1 \cdot w_0 \cdot E_n$$

$$y_0 = \overline{w_1} \cdot \overline{w_0} \cdot I$$
$$y_1 = \overline{w_1} \cdot w_0 \cdot I$$
$$y_2 = w_1 \cdot \overline{w_0} \cdot I$$
$$y_3 = w_1 \cdot w_0 \cdot I$$

$$y_0 = \overline{w_1} \cdot \overline{w_0} \cdot I$$
$$y_1 = \overline{w_1} \cdot w_0 \cdot I$$
$$y_2 = w_1 \cdot \overline{w_0} \cdot I$$
$$y_3 = w_1 \cdot w_0 \cdot I$$

HW

- Design full subtractor using 3:8 decoder
- Design full subtractor using 1:8 demultiplexer
- BCD to 7-segment decoder