Digital Circuits

ECS 371

Dr. Prapun Suksompong

prapun@siit.tu.ac.th

Lecture 12

Office Hours:
BKD 3601-7
Monday 9:00-10:30, 1:30-3:30
Tuesday 10:30-11:30

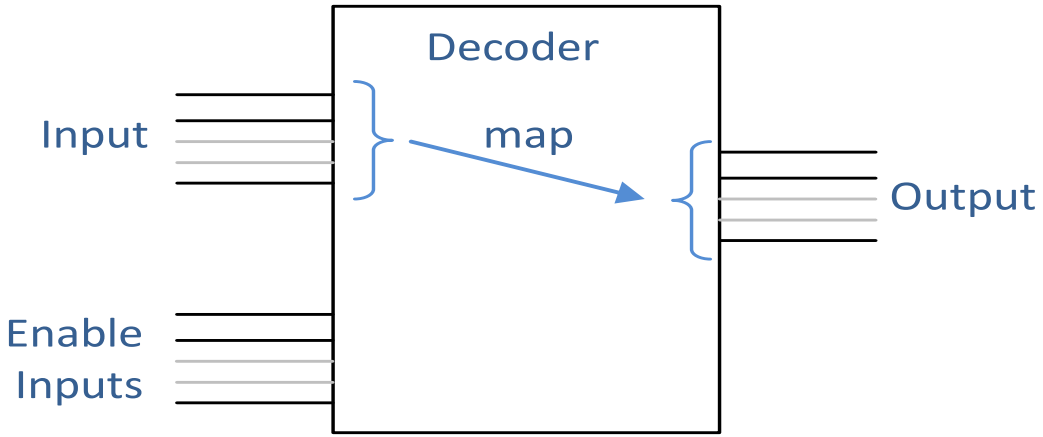
ECS371.PRAPUN.COM

Announcement

- No new HW.
- Reading Assignment
  - Chapter 6: 6-5, 6-8, 6-9

Review: BIN/DEC Decoder

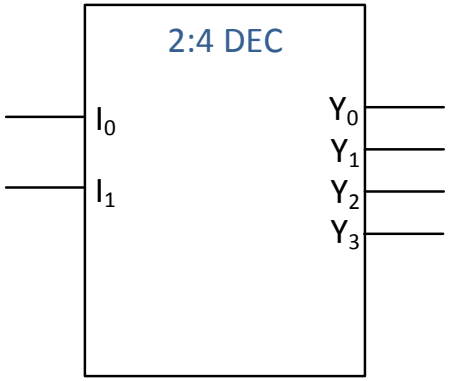
- Many output lines.
- Only one of the output line is asserted at any time.
- To find out which output line is asserted, consider the inputs as one binary number. Convert this binary number to a decimal number. This decimal number tells which output line is asserted by the inputs.



Truth Tables of Basic Decoder

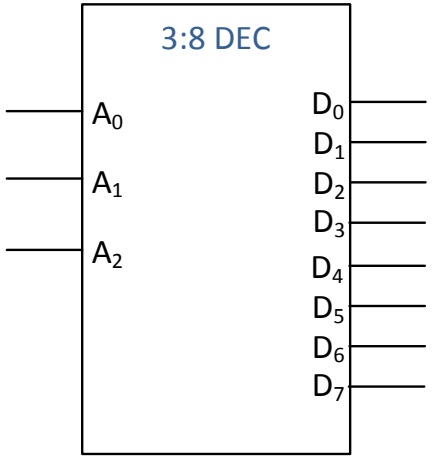
2:4 DEC

Input		Output			
I_1	I_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0



3:8 DEC

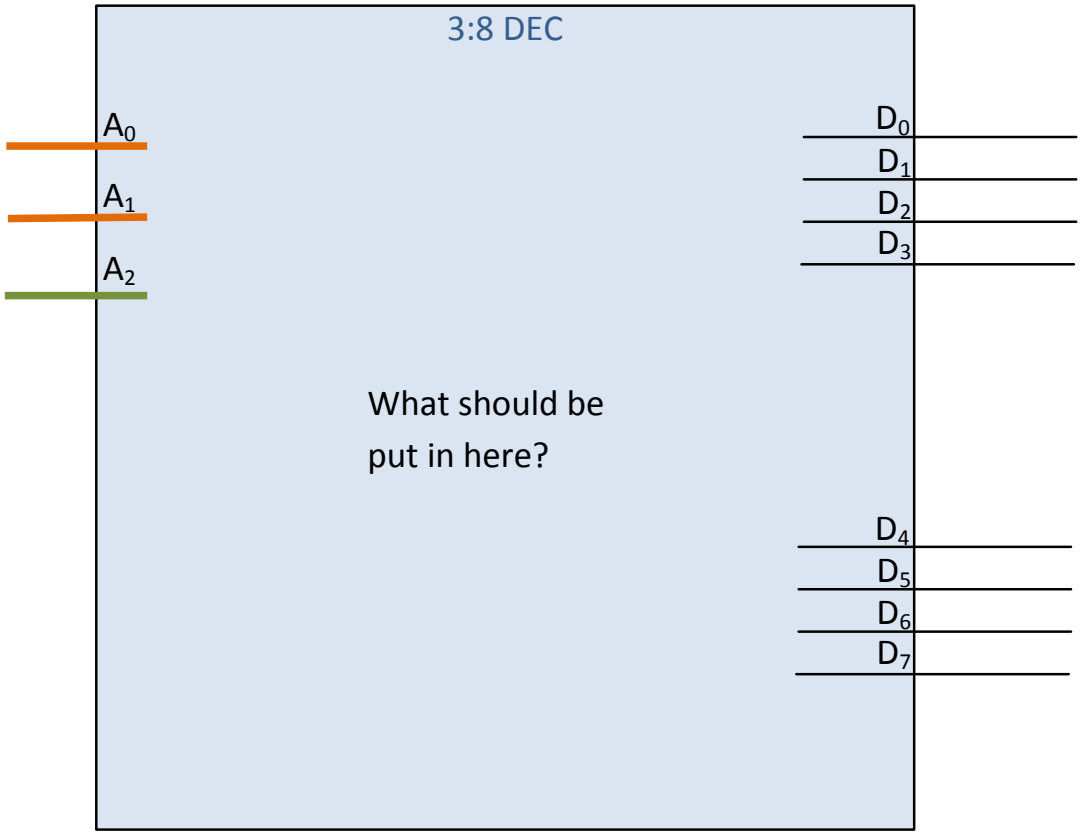
Input			Output							
A_2	A_1	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0



Decoder Expansion

How can we build a larger decoder from smaller ones?

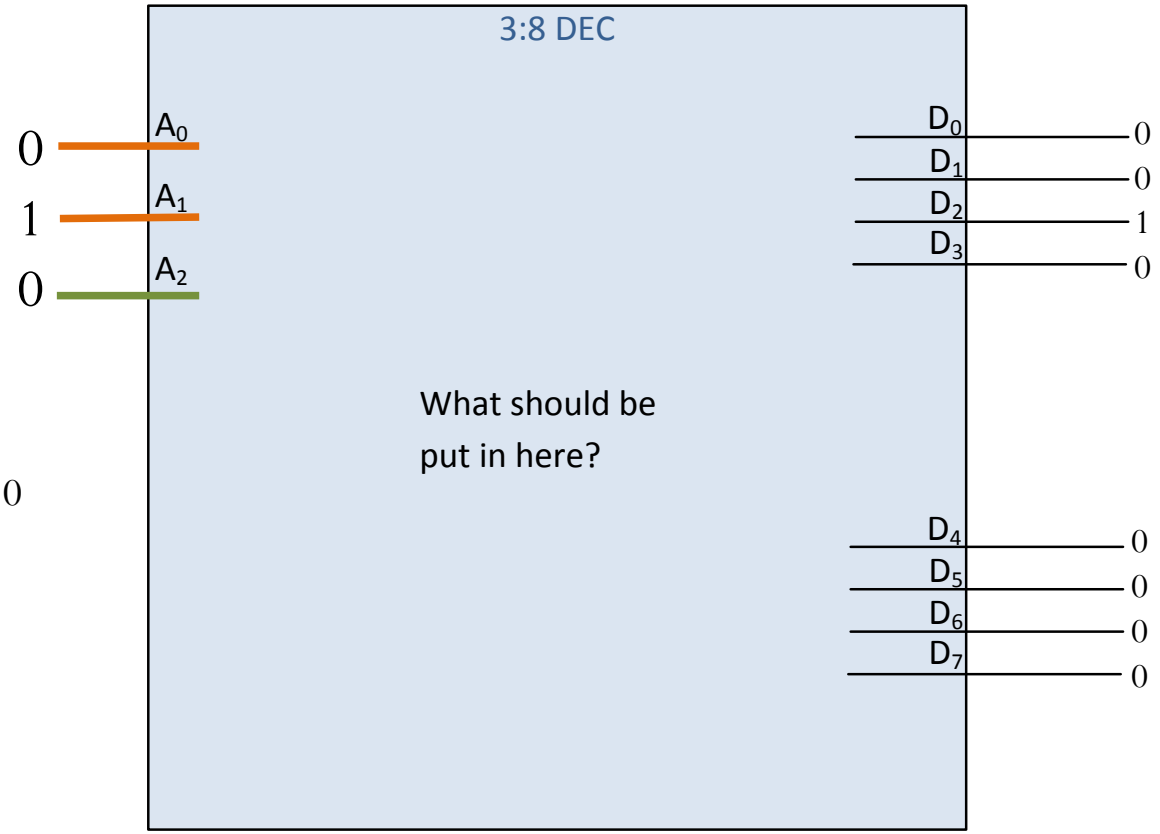
Construct a 3-to-8 decoder from two 2-to-4 decoders



Decoder Expansion

How can we build a larger decoder from smaller ones?

Construct a 3-to-8 decoder from two 2-to-4 decoders

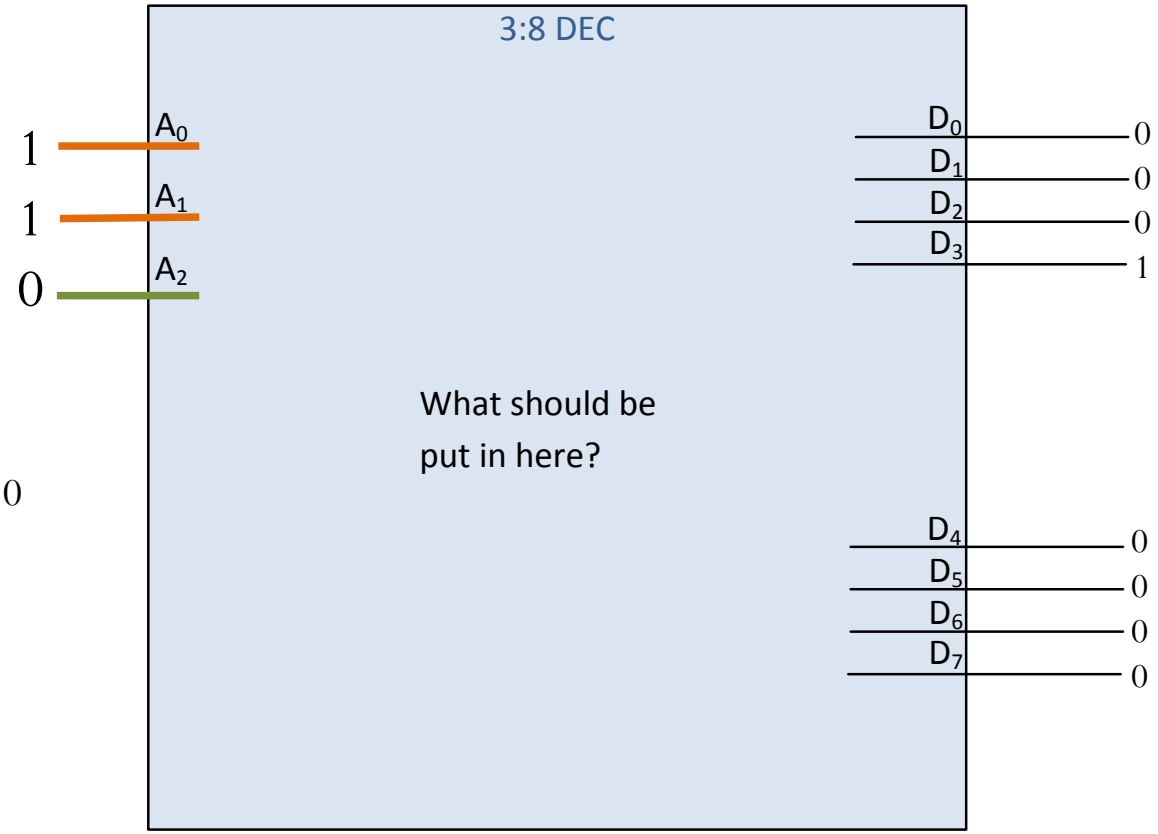


$010_2 = 2_{10}$

Decoder Expansion

How can we build a larger decoder from smaller ones?

Construct a 3-to-8 decoder from two 2-to-4 decoders

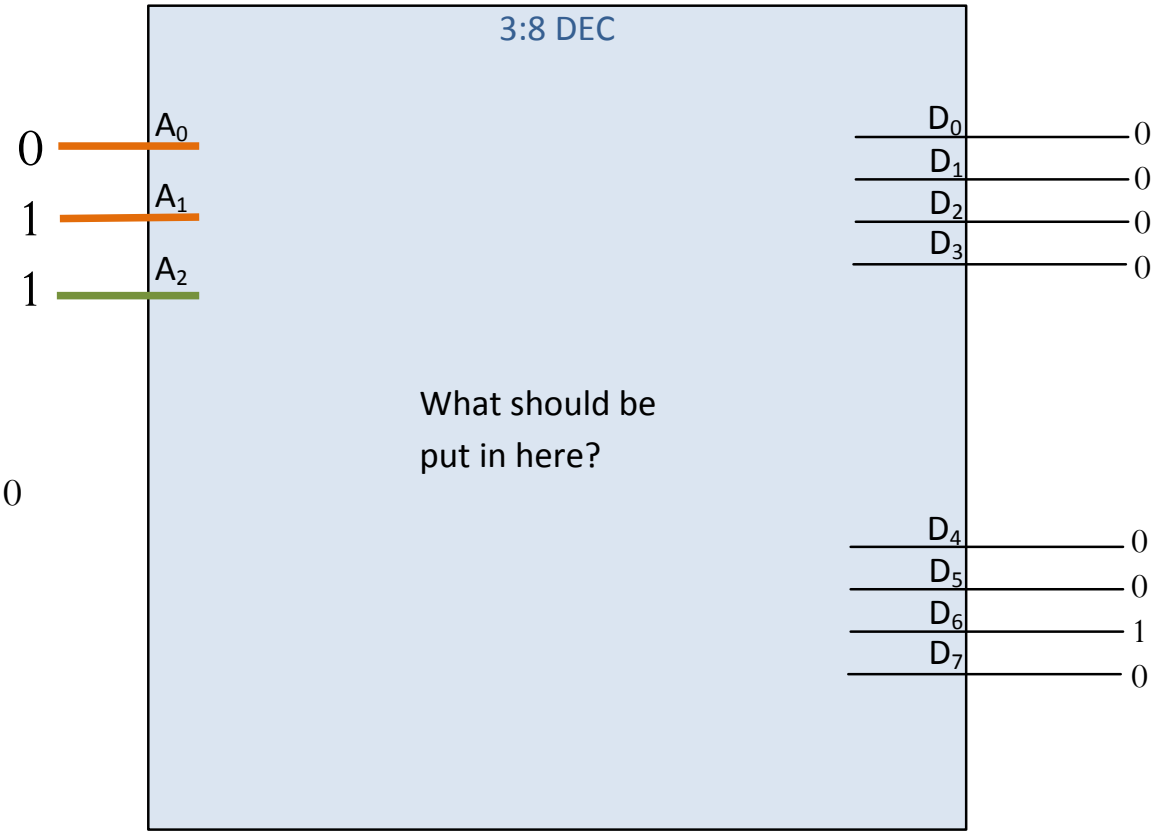


$011_2 = 3_{10}$

Decoder Expansion

How can we build a larger decoder from smaller ones?

Construct a 3-to-8 decoder from two 2-to-4 decoders

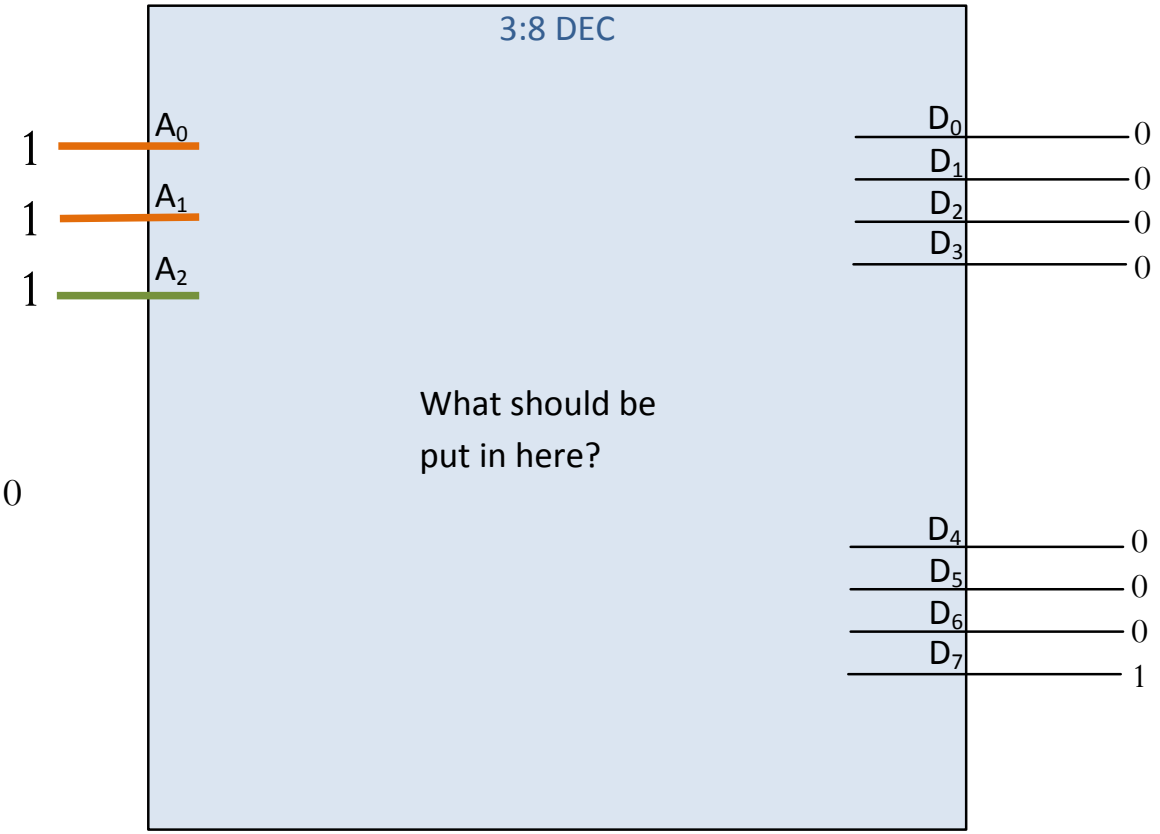


$110_2 = 6_{10}$

Decoder Expansion

How can we build a larger decoder from smaller ones?

Construct a 3-to-8 decoder from two 2-to-4 decoders



$111_2 = 7_{10}$

Decoder Expansion

Construct a 3-to-8 decoder from two 2-to-4 decoders:

Observe that the truth table of the 3:8 DEC contains two truth tables of the 2:4 DEC.

3:8 DEC

Input			Output							
A_2	A_1	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

2:4 DEC

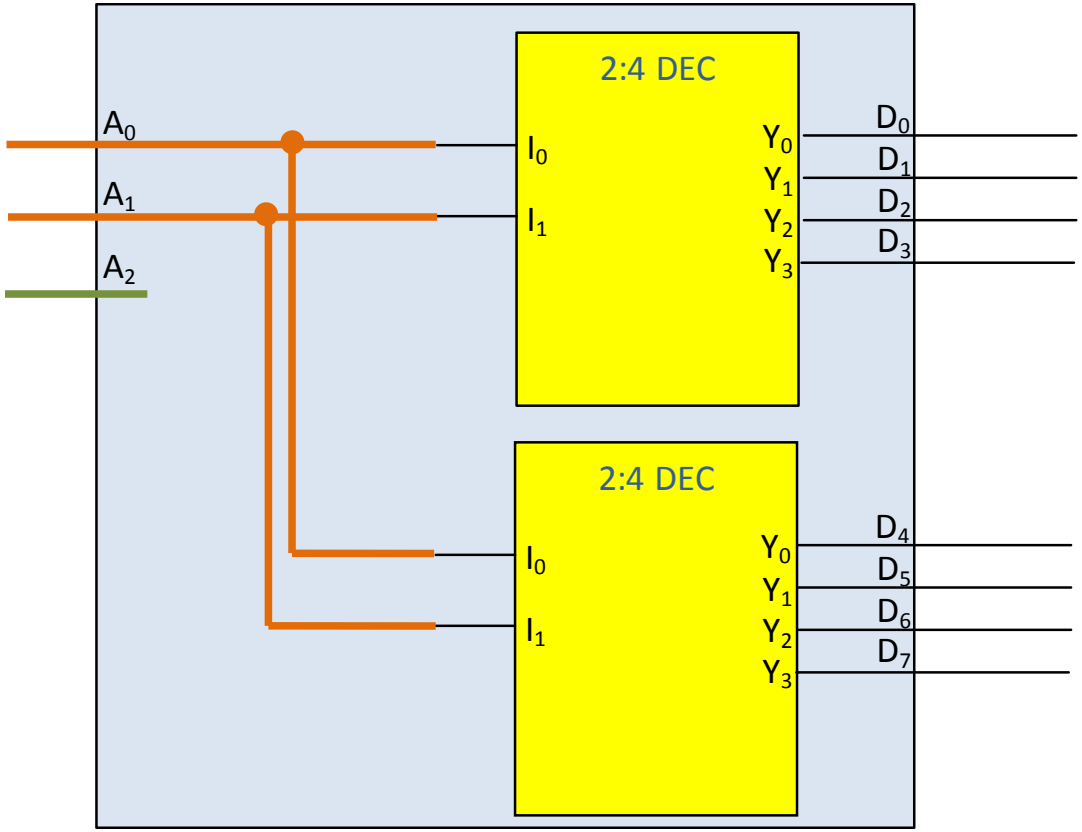
Input		Output			
I_1	I_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Two large areas of 0s (negated)

These areas correspond to A_2.

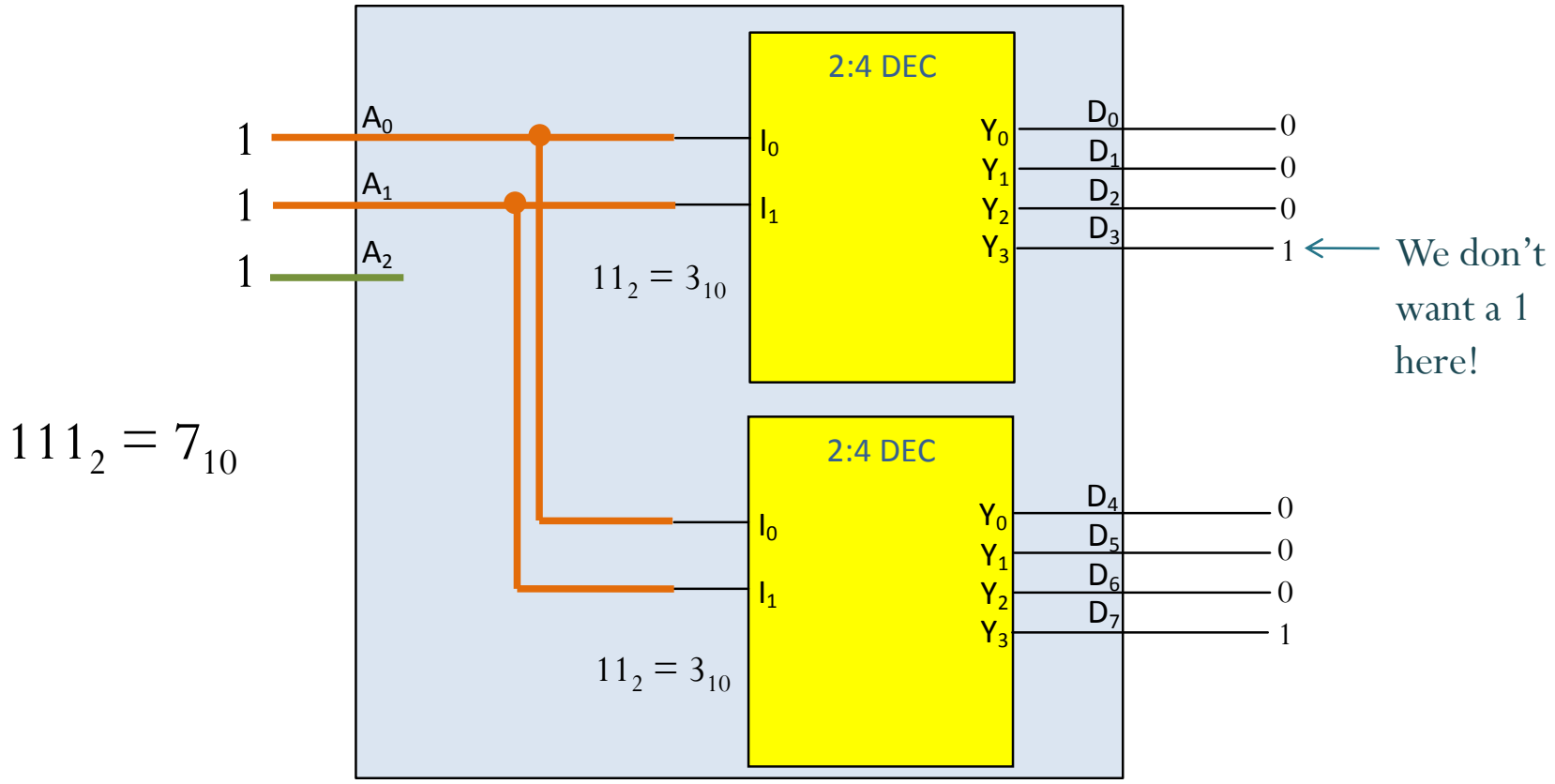
Decoder Expansion

Construct a 3-to-8 decoder from two 2-to-4 decoders:



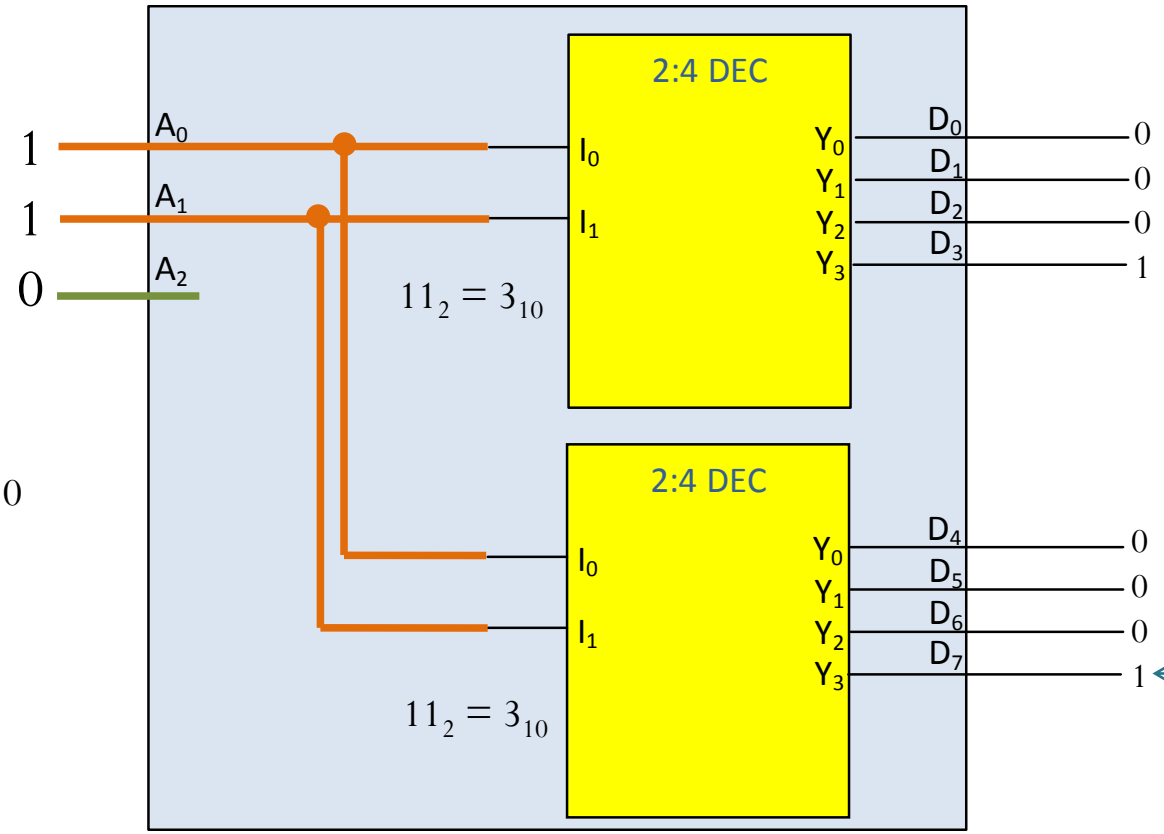
Decoder Expansion

Construct a 3-to-8 decoder from two 2-to-4 decoders:



Decoder Expansion

Construct a 3-to-8 decoder from two 2-to-4 decoders:

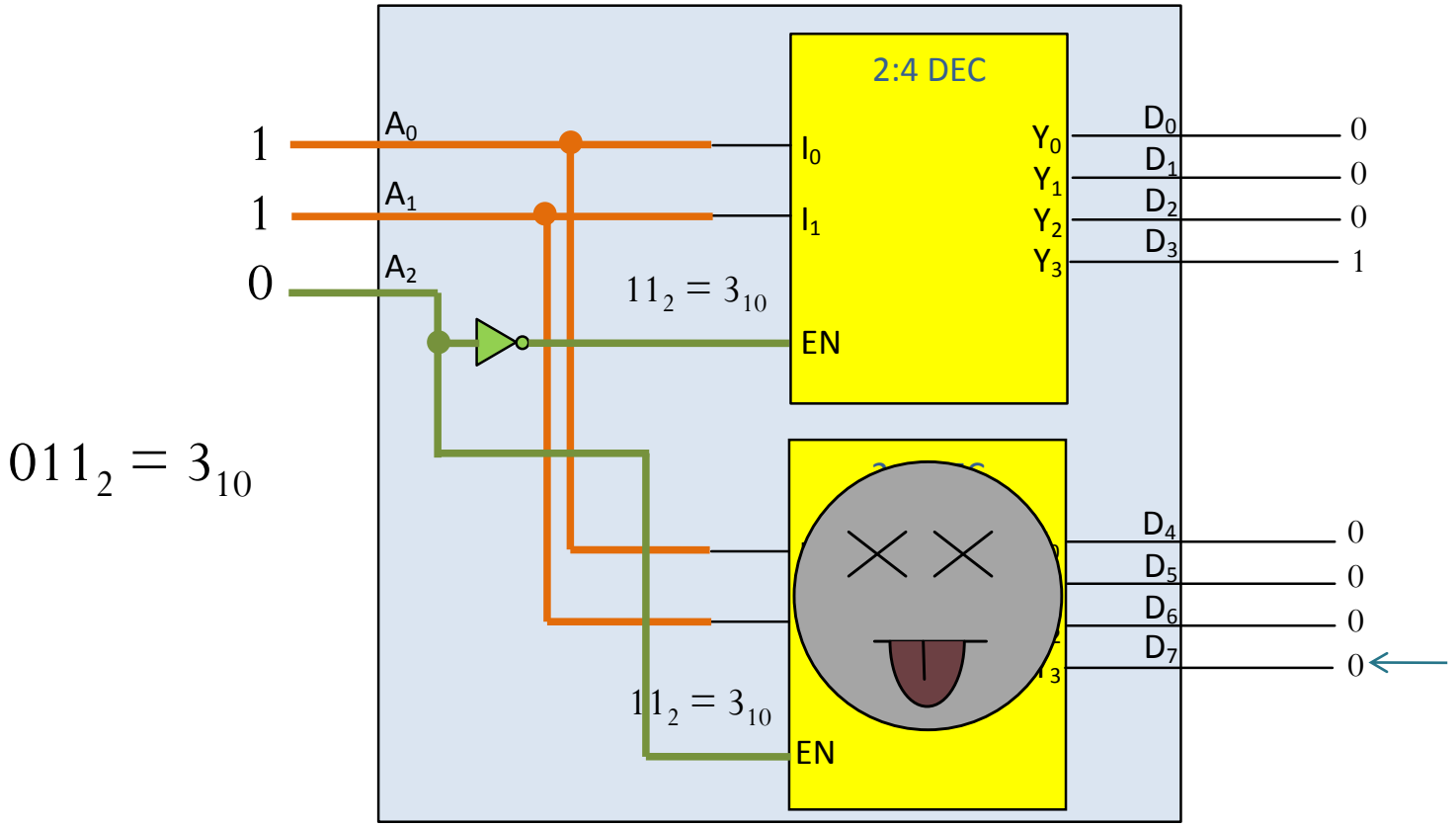


$011_2 = 3_{10}$

We don't want a 1 here!

Decoder Expansion

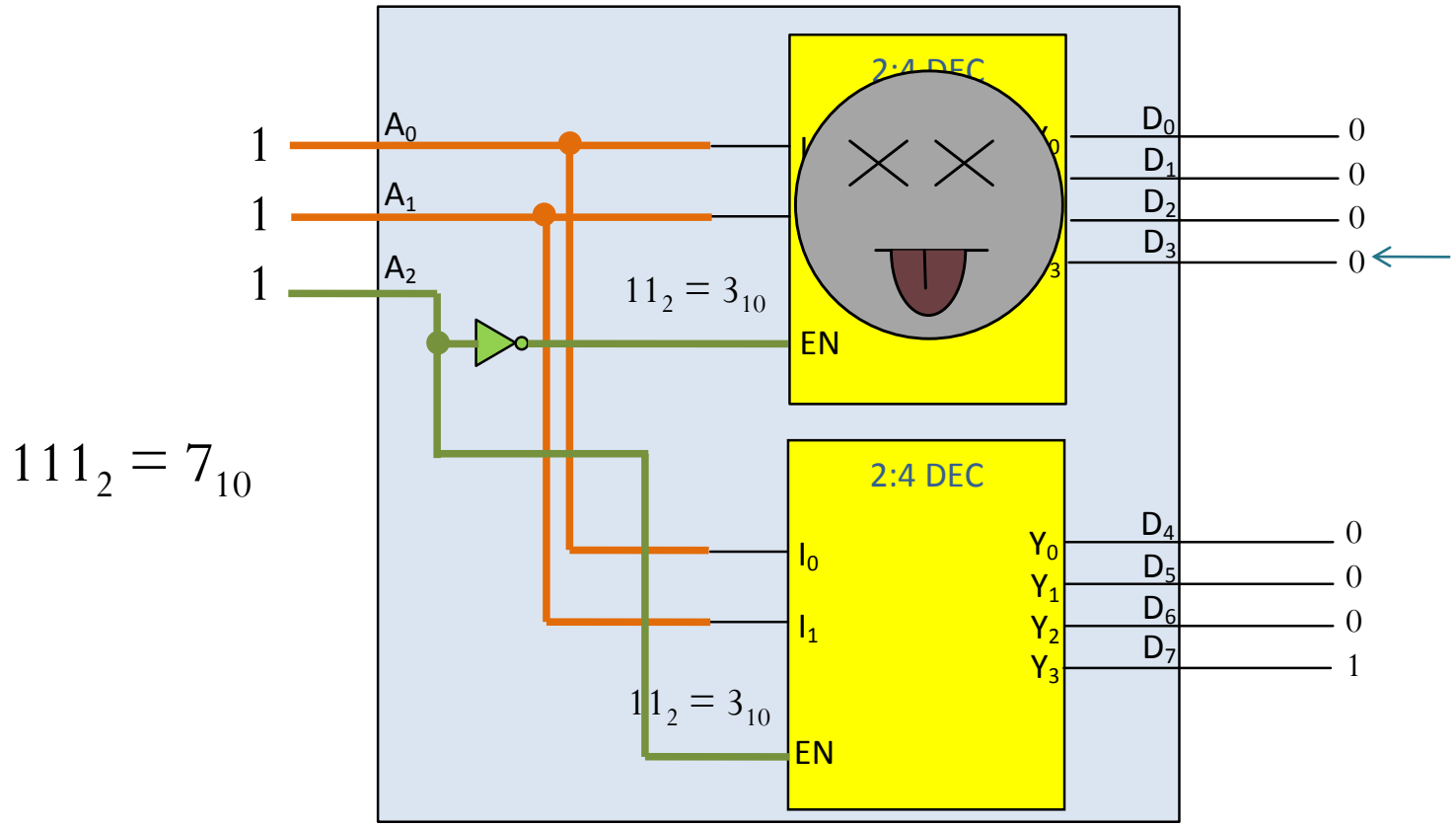
A_2 is connected to the EN of each 2:4 decoder to choose the 1 that we want.



$011_2 = 3_{10}$

Decoder Expansion

A_2 is connected to the EN of each 2:4 decoder to choose the 1 that we want.



$111_2 = 7_{10}$

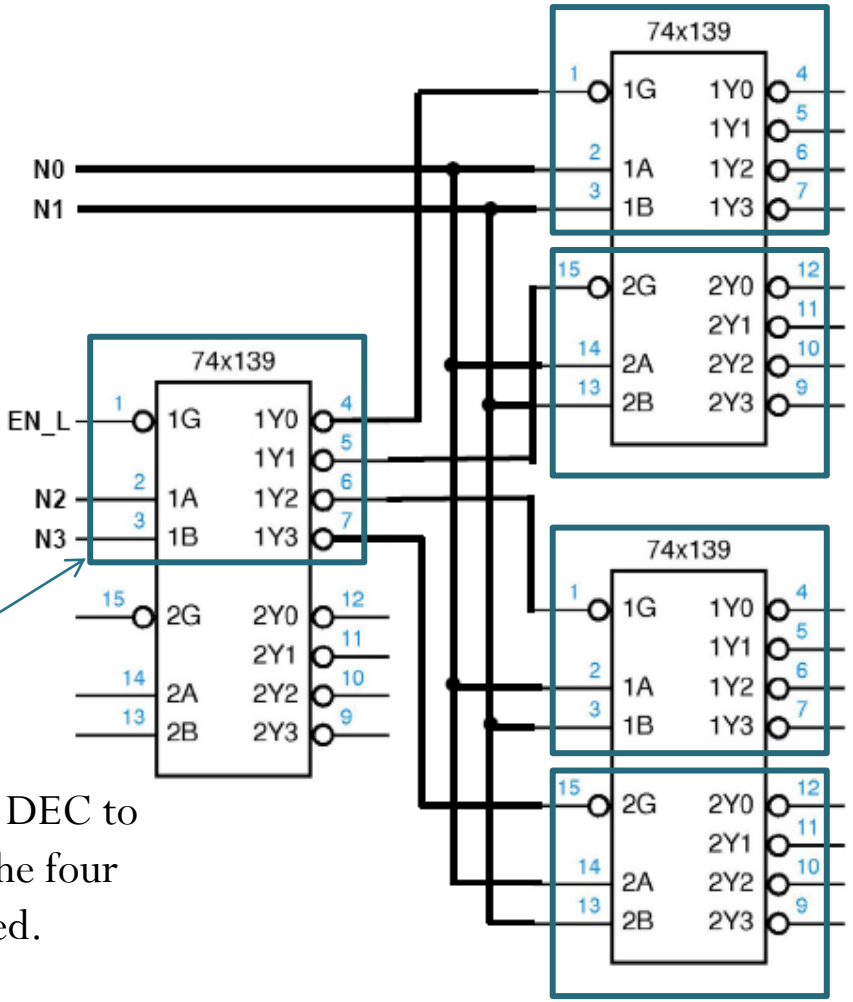
Decoder Expansion: Summary

- To increase the input by one bit (2:4 to 3:8)
  - Use two smaller decoders.
  - Connect the lower significant bits to both decoders.
  - Use the MSB to control which decoder is enabled.
- To increase the input by two bits (2:4 to 4:16)
  - Start with four smaller decoders.
  - Connect the lower significant bits to all four decoders.
  - Use the two higher significant bits to control which decoder is enabled.

Example

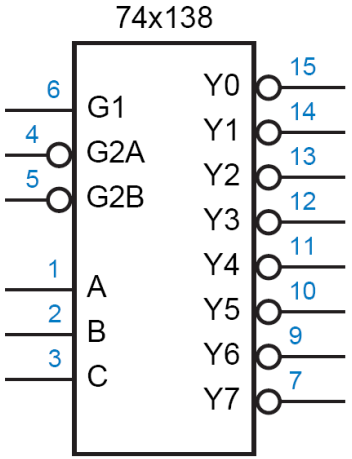
Construct a 4:16 decoder with an active-LOW enable from three 74x139.

Four 2:4 DEC in this column.



Use one more 2:4 DEC to control which of the four decoders is enabled.

74x138: 3:8 Decoder

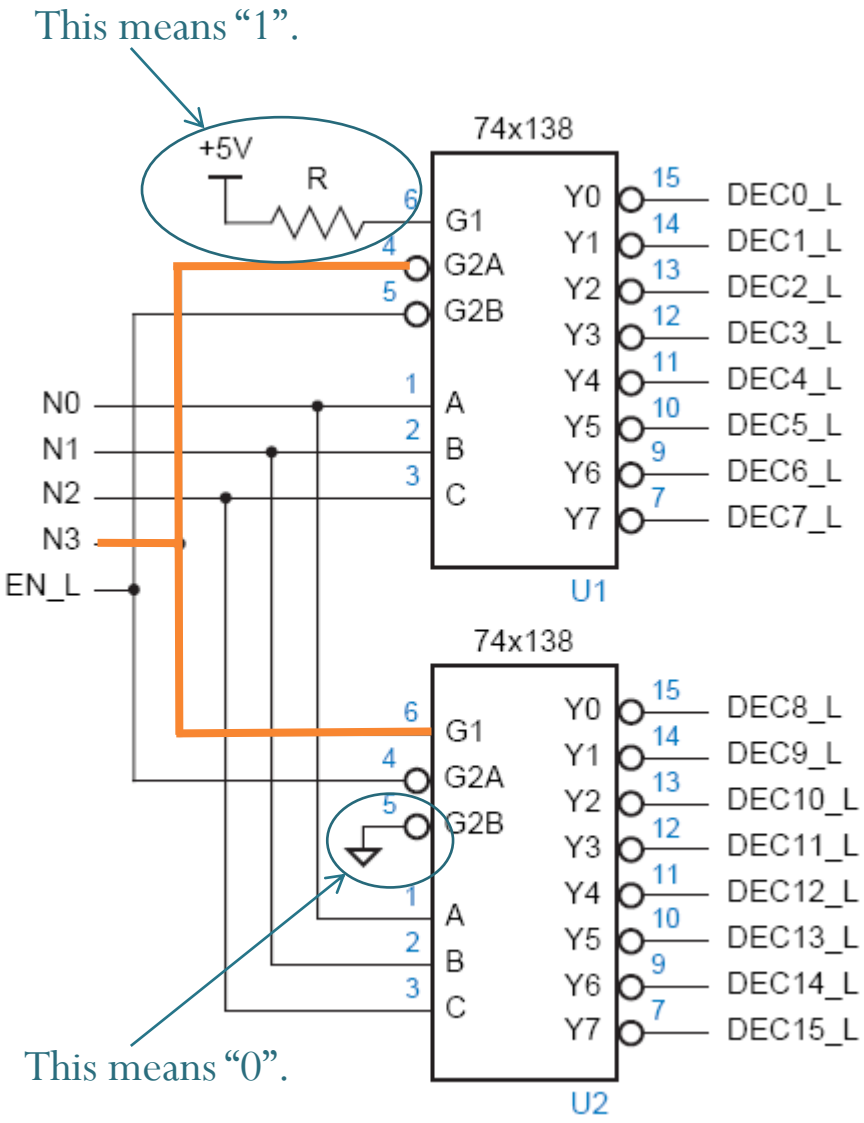


- Active-LOW outputs
- Three enable inputs.

Inputs						*Outputs*							
G1	G2A_L	G2B_L	C	B	A	Y7_L	Y6_L	Y5_L	Y4_L	Y3_L	Y2_L	Y1_L	Y0_L
0	x	x	x	x	x	1	1	1	1	1	1	1	1
x	1	x	x	x	x	1	1	1	1	1	1	1	1
x	x	1	x	x	x	1	1	1	1	1	1	1	1
1	0	0	0	0	0	1	1	1	1	1	1	1	0
1	0	0	0	0	1	1	1	1	1	1	1	0	1
1	0	0	0	1	0	1	1	1	1	1	0	1	1
1	0	0	0	1	1	1	1	1	1	0	1	1	1
1	0	0	1	0	0	1	1	1	0	1	1	1	1
1	0	0	1	0	1	1	1	0	1	1	1	1	1
1	0	0	1	1	0	1	0	1	1	1	1	1	1
1	0	0	1	1	1	0	1	1	1	1	1	1	1

Example

Construct a 4:16 decoder with an active-LOW enable (EN) from two 74x138 decoder.

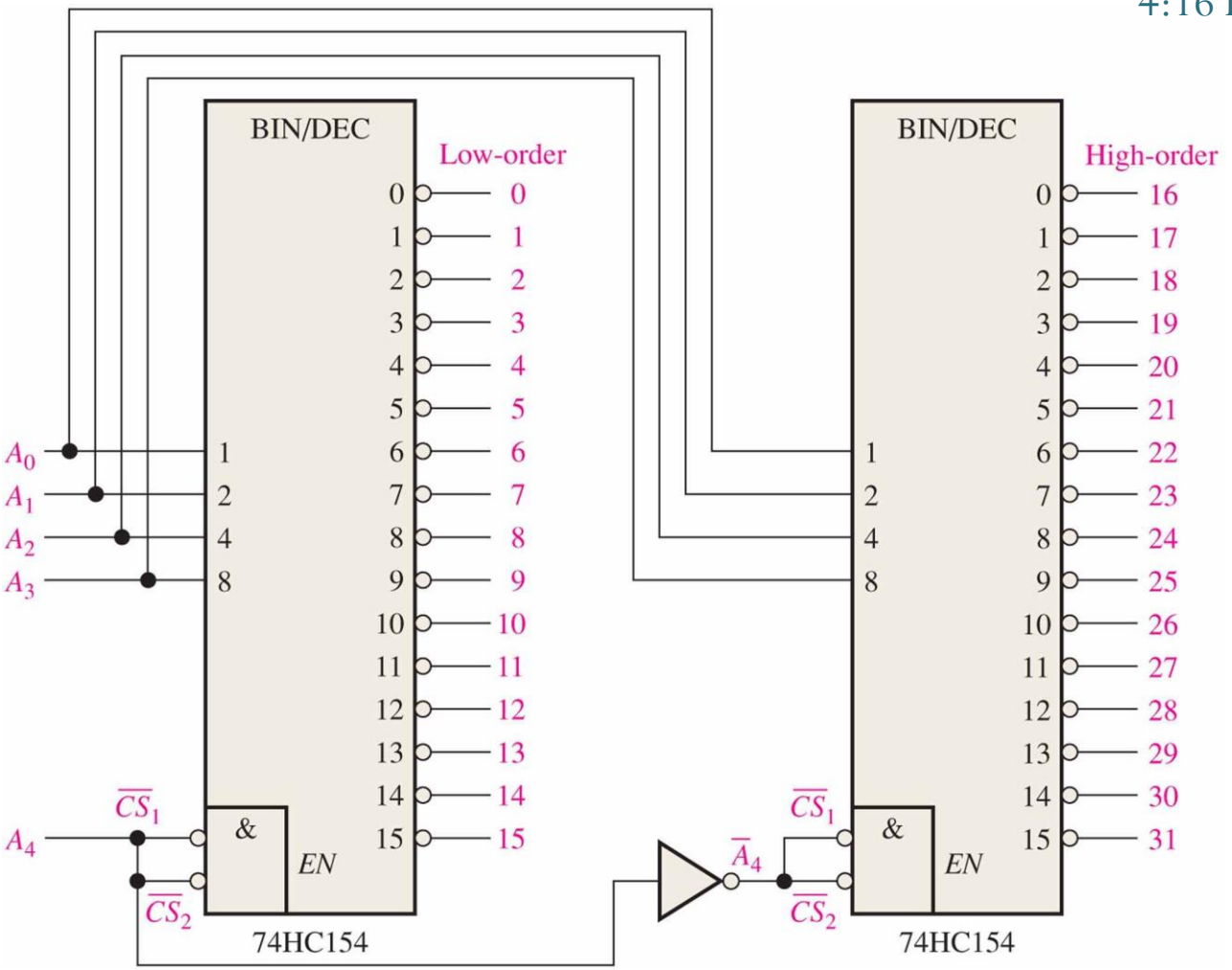


Because the 74x138 have both active-LOW EN and active-HIGH EN, we can use the extra bubble to replace the extra NOT gate.

Example

Construct a 5:32 decoder with active-LOW outputs from two 74x154 and one inverter.

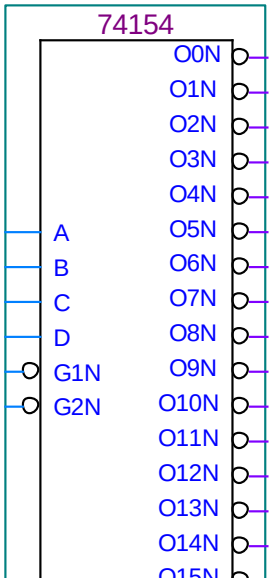
4:16 DEC



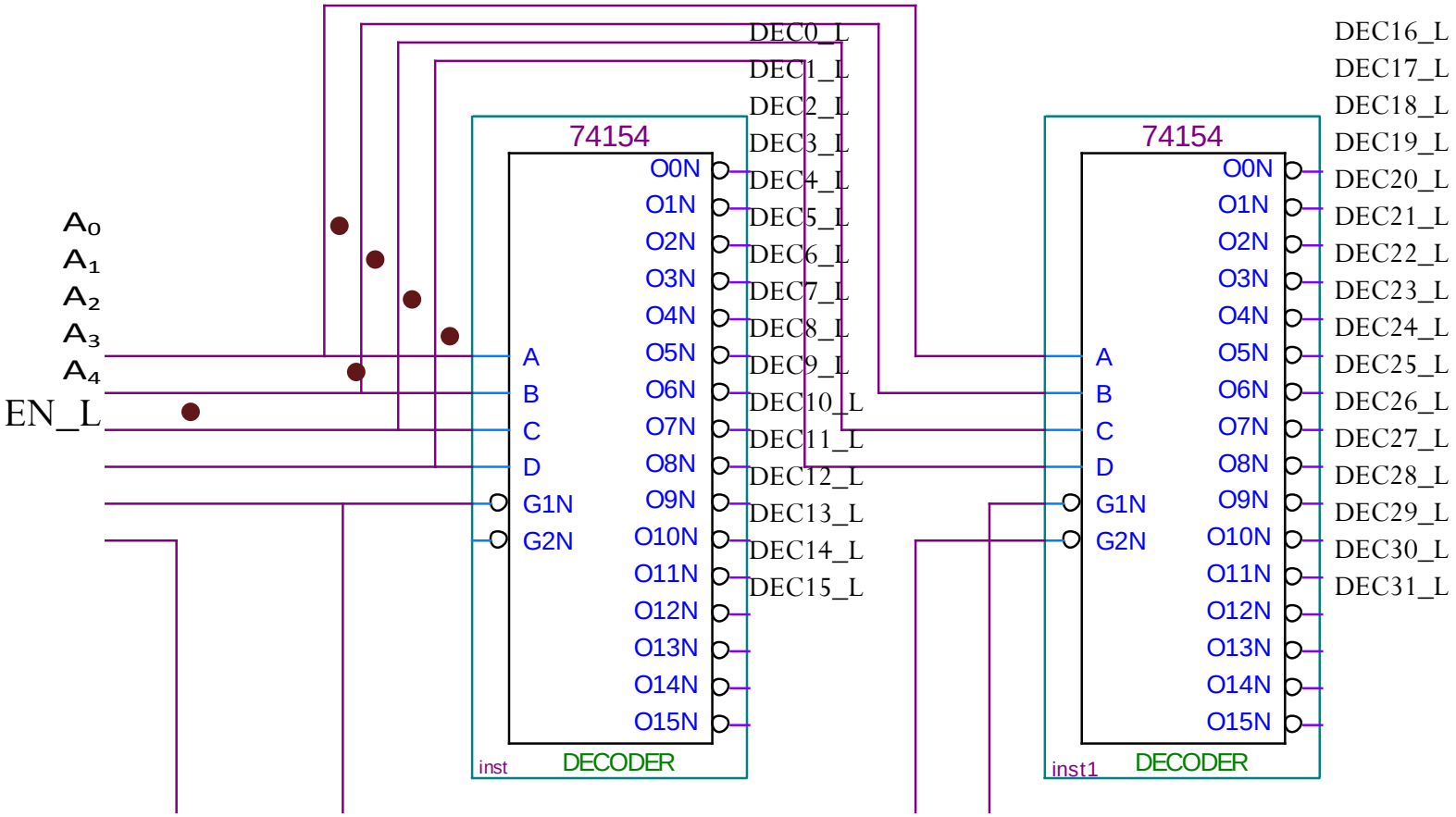
(Midterm: July 30!)

Exercise (Sample Exam Problem)

- Construct a 5:32 decoder with active-LOW outputs and one active-LOW EN.
- Use two 74x154 and one inverter.



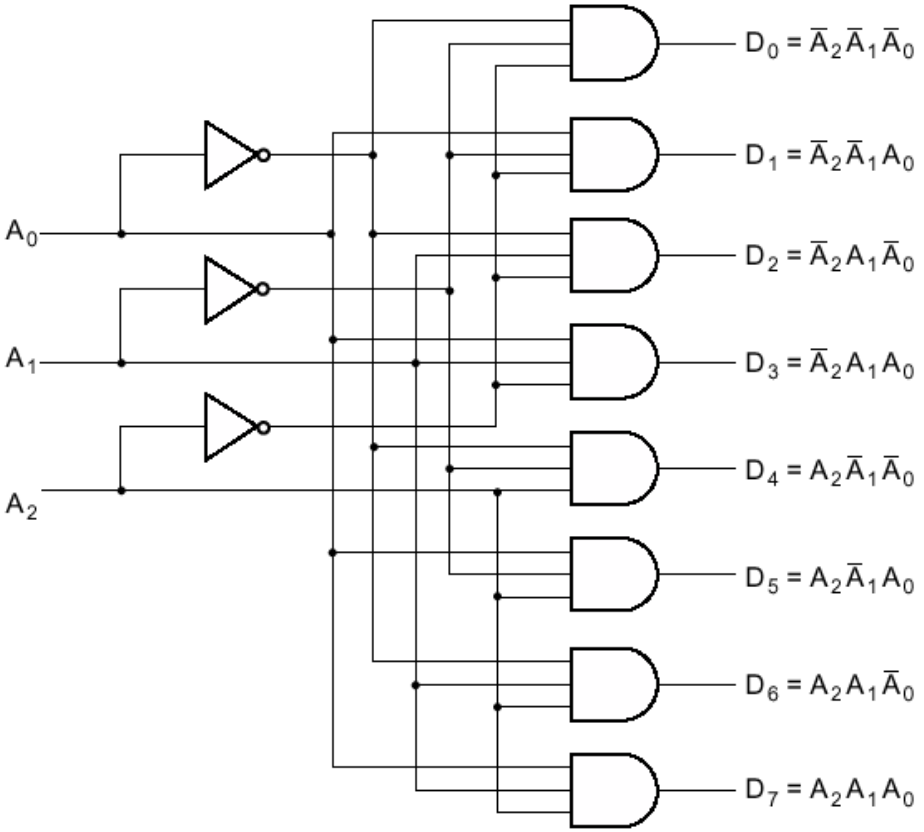
Solution


A0
A1
A2
A3
A4
EN_L
74154
A
B
C
D
G1N
G2N
O0N
O1N
O2N
O3N
O4N
O5N
O6N
O7N
O8N
O9N
O10N
O11N
O12N
O13N
O14N
O15N
inst
DECODER
DEC0_L
DEC1_L
DEC2_L
DEC3_L
DEC4_L
DEC5_L
DEC6_L
DEC7_L
DEC8_L
DEC9_L
DEC10_L
DEC11_L
DEC12_L
DEC13_L
DEC14_L
DEC15_L
74154
A
B
C
D
G1N
G2N
O0N
O1N
O2N
O3N
O4N
O5N
O6N
O7N
O8N
O9N
O10N
O11N
O12N
O13N
O14N
O15N
inst1
DECODER
DEC16_L
DEC17_L
DEC18_L
DEC19_L
DEC20_L
DEC21_L
DEC22_L
DEC23_L
DEC24_L
DEC25_L
DEC26_L
DEC27_L
DEC28_L
DEC29_L
DEC30_L
DEC31_L

Decoder as General Purpose Logic

Any combinational circuit with n inputs and m outputs can be implemented with an n-to-2^n-line decoder and m OR gate

Observe that the 3:8 decoder generates all possible minterms.

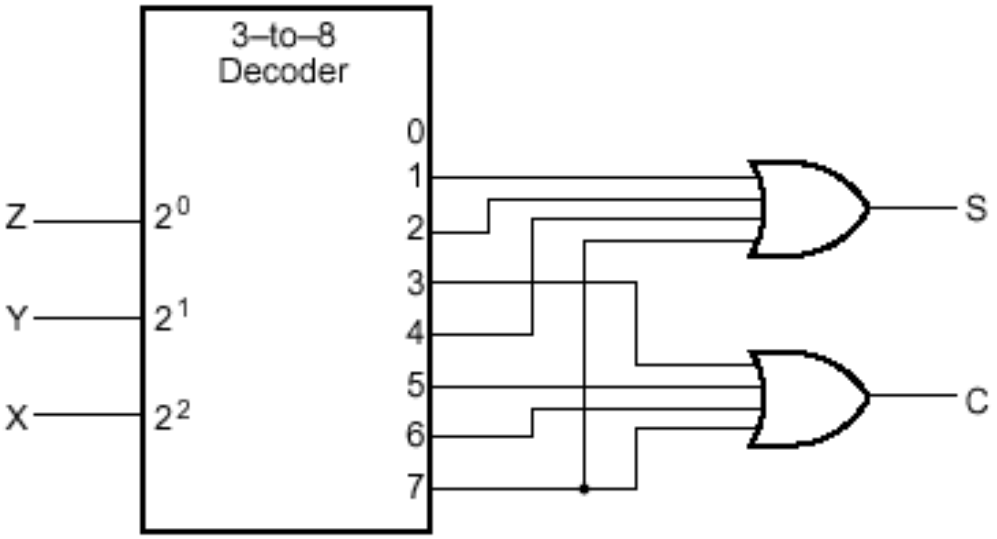


Example

Implement a full adder circuit with a decoder and OR gates

- $S = \sum_{X,Y,Z}(1,2,4,7)$
- $C = \sum_{X,Y,Z}(3,5,6,7)$

Inputs			Outputs	
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

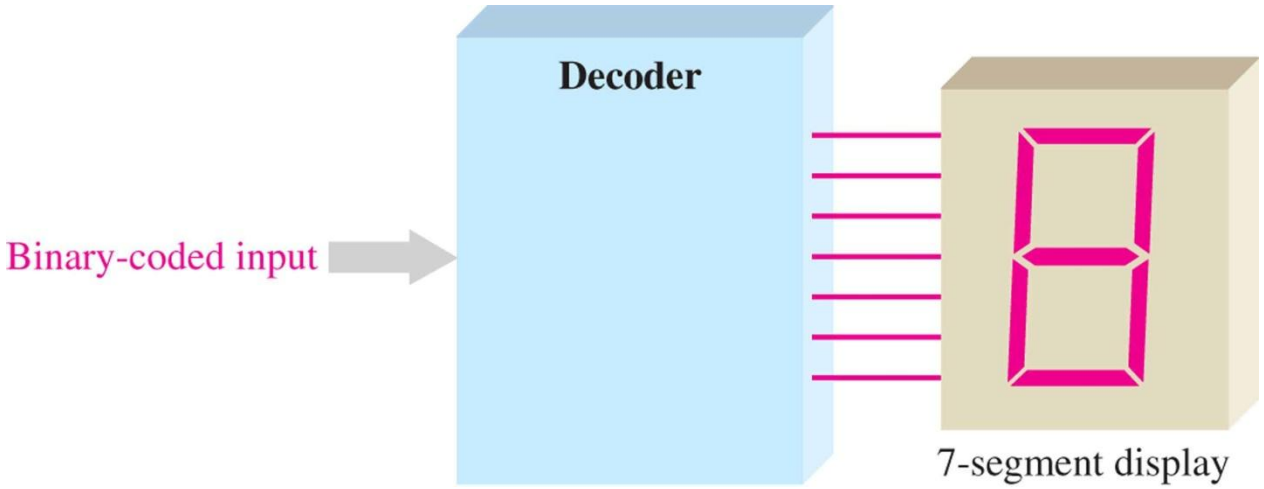


Note: If the decoder's output is active-LOW, then we use NAND gates instead of the OR gates.

Other Decoders

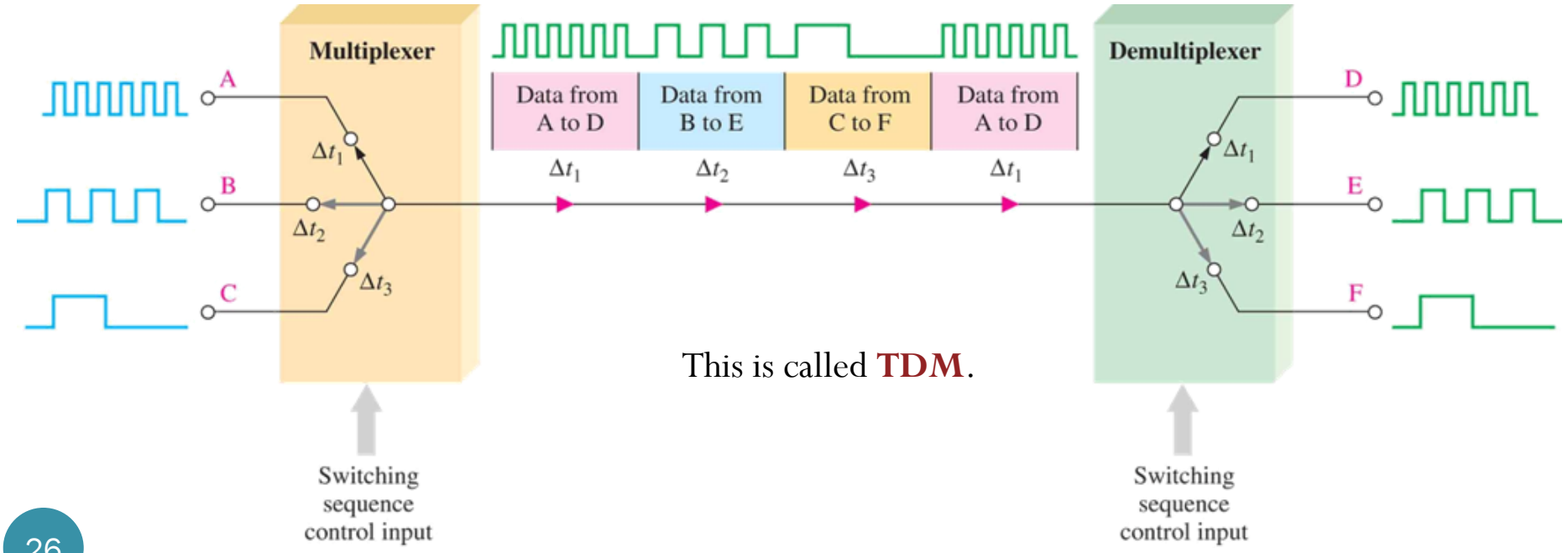
In general, a decoder converts coded information, such as binary number, into non-coded form.

Later, (if time permitted) we will talk about other types of decoder.



Multiplexing/Demultiplexing

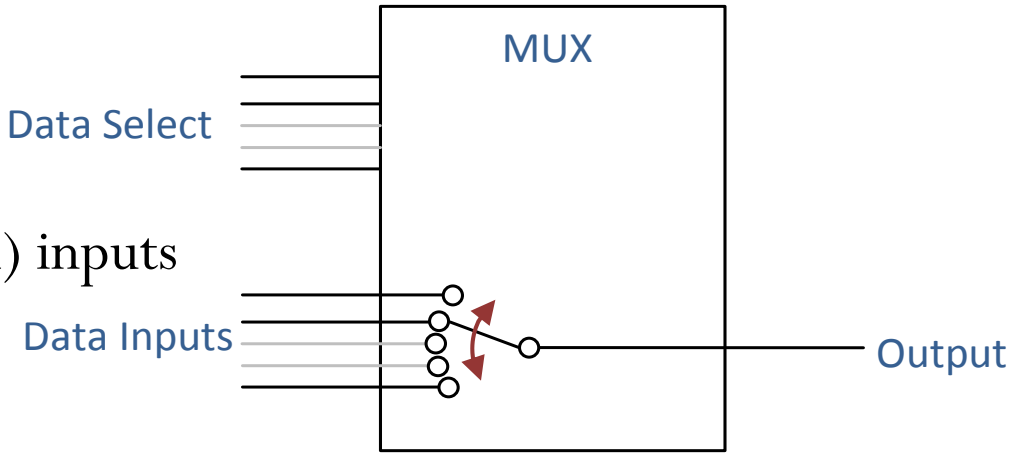
- The **multiplexer**, or **mux** for short, is a logic circuit that switches digital data from several input lines onto a single output line in a specified time sequence.
- The **demultiplexer (demux)** is a logic circuit that switches digital data from one input line to several output lines in a specified time sequence



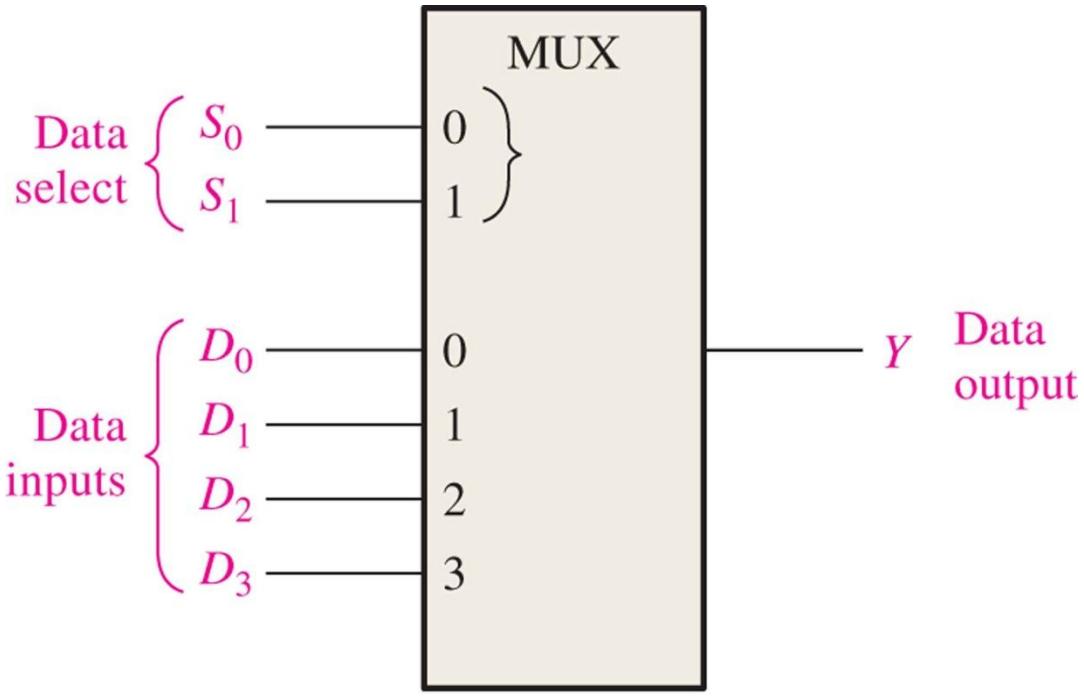
This is called **TDM**.

Multiplexer (Data Selector)

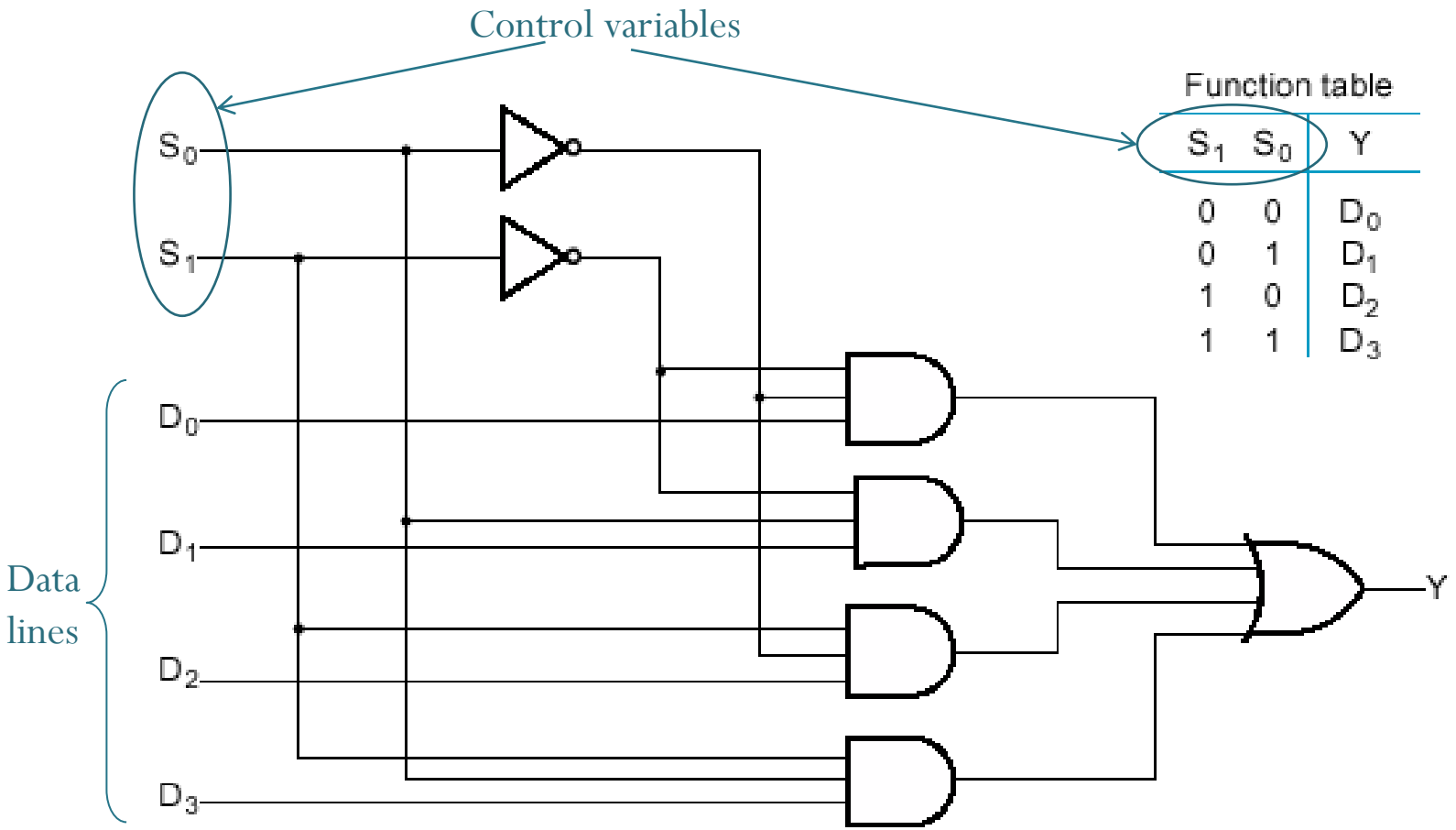
- Select binary information from one of many input lines and directs the information to a single output line.
- Allow digital information from several sources to be routed onto a single line for transmission over that line to a common destination.
- Basic multiplexer has
  1. Data-input lines
  2. Single output line.
  3. Data-select (control) inputs



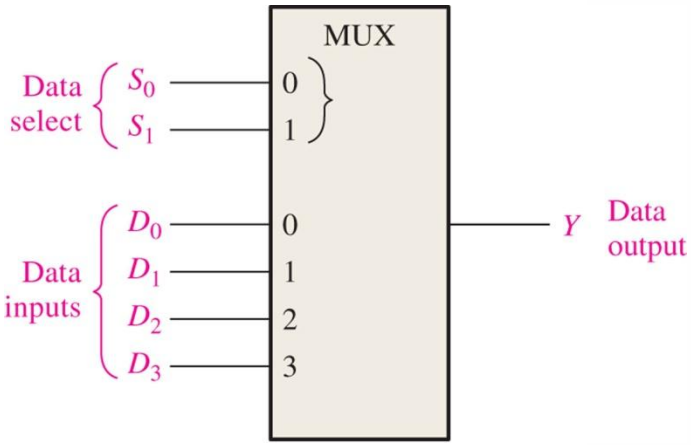
Example: 4-to-1-line multiplexer



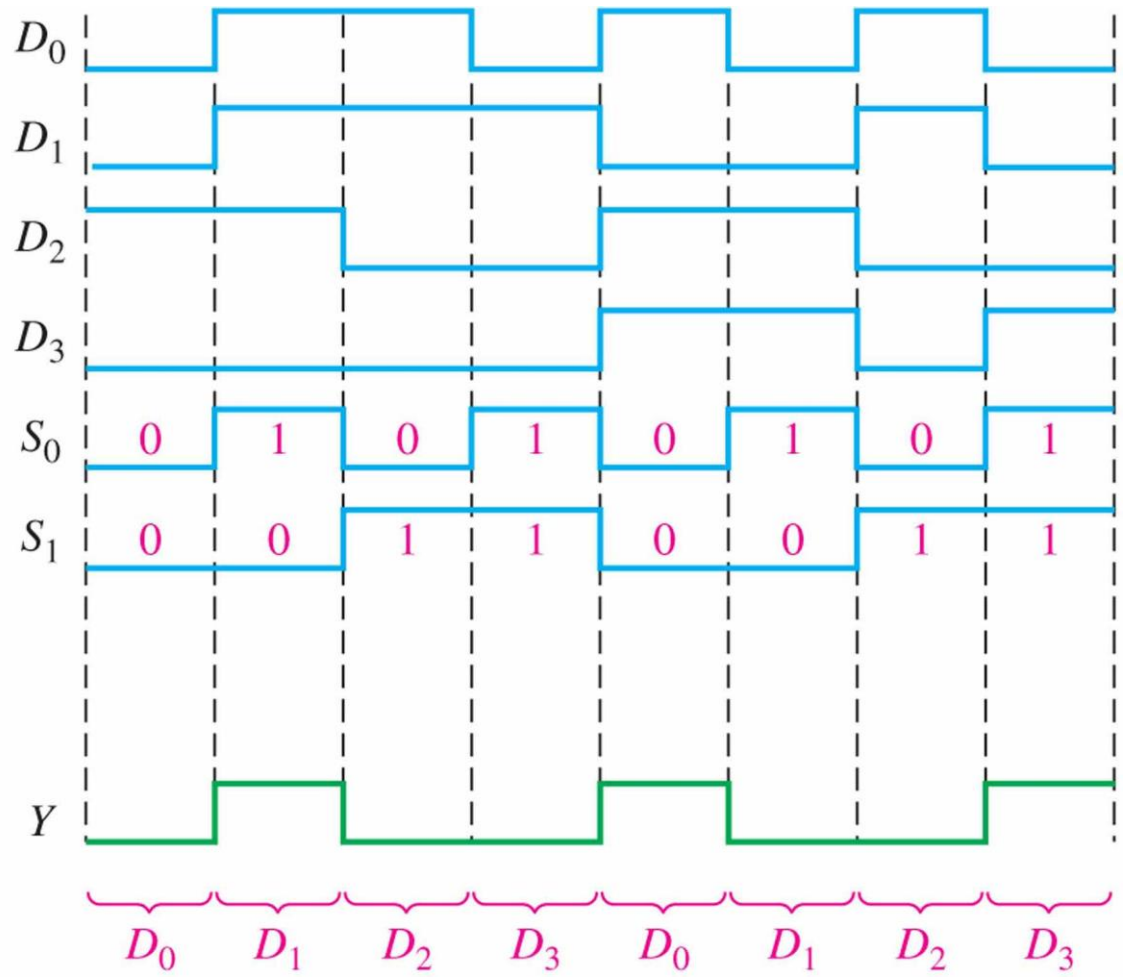
4:1 MUX: Logic Diagram & Truth Table



Example


MUX
Data select
S_0
S_1
0
1
Data inputs
D_0
D_1
D_2
D_3
0
1
2
3
Y
Data output

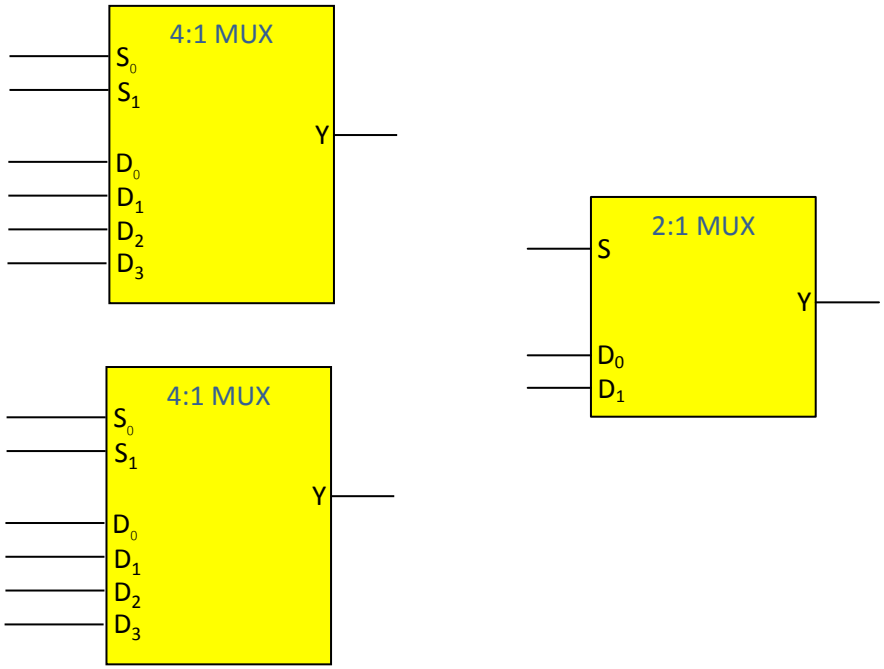
(a)


D_0
D_1
D_2
D_3
S_0 0 1 0 1 0 1 0 1
S_1 0 0 1 1 0 0 1 1
Y
D_0 D_1 D_2 D_3 D_0 D_1 D_2 D_3

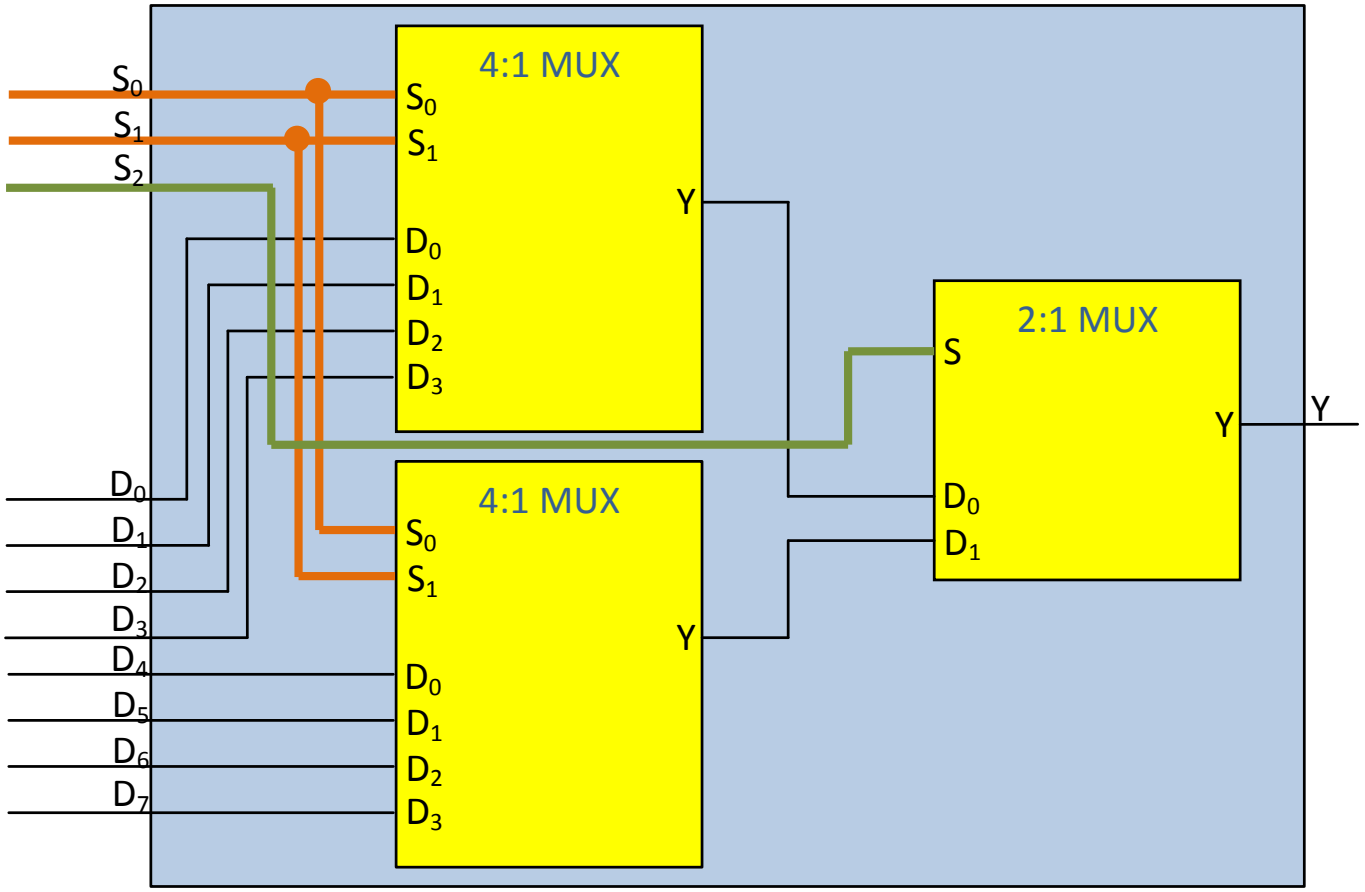
(b)

Multiplexer Expansion

Use two 4:1 MUXs and one 2:1 to create 8:1 MUX.

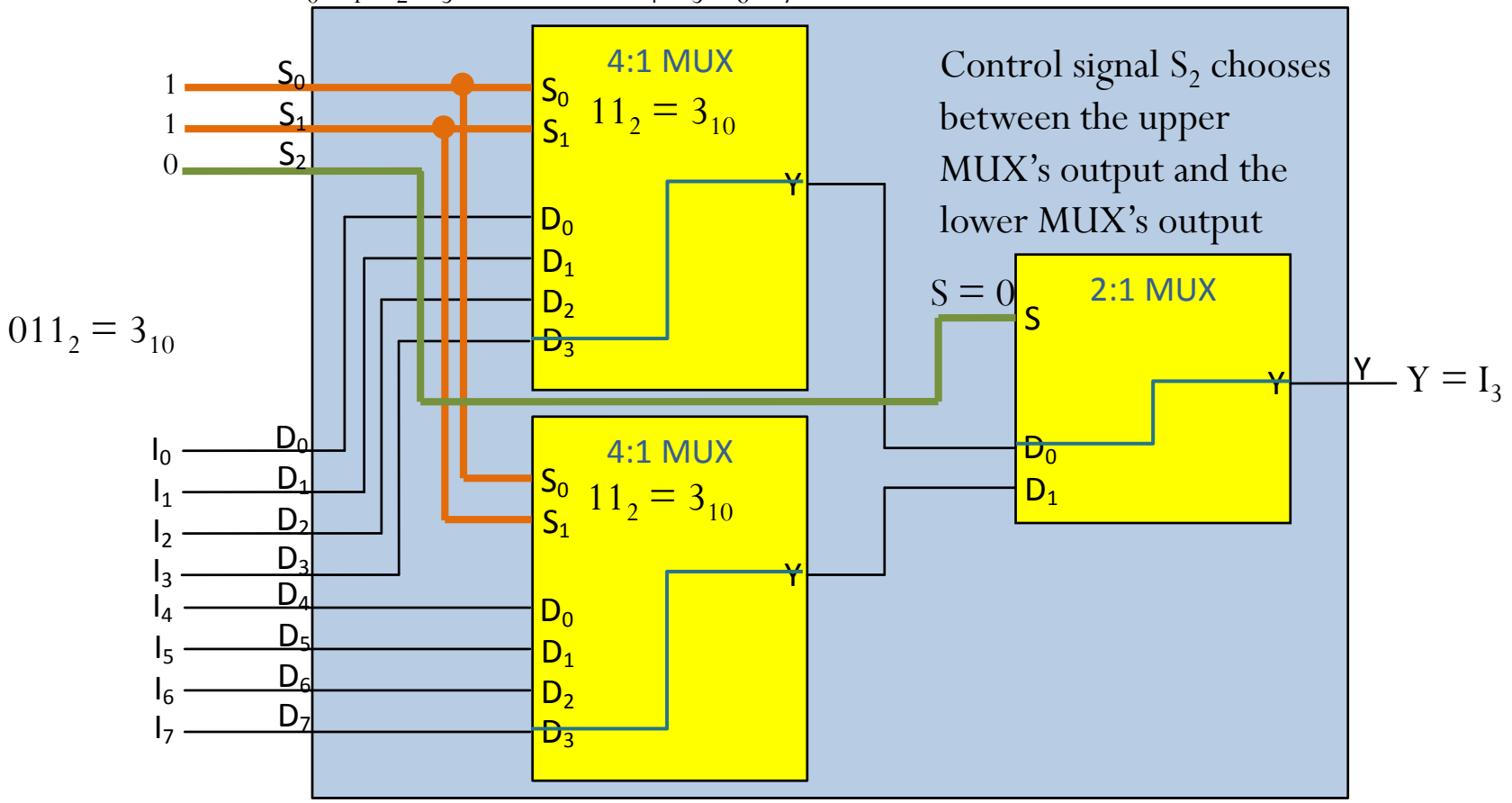


Multiplexer Expansion



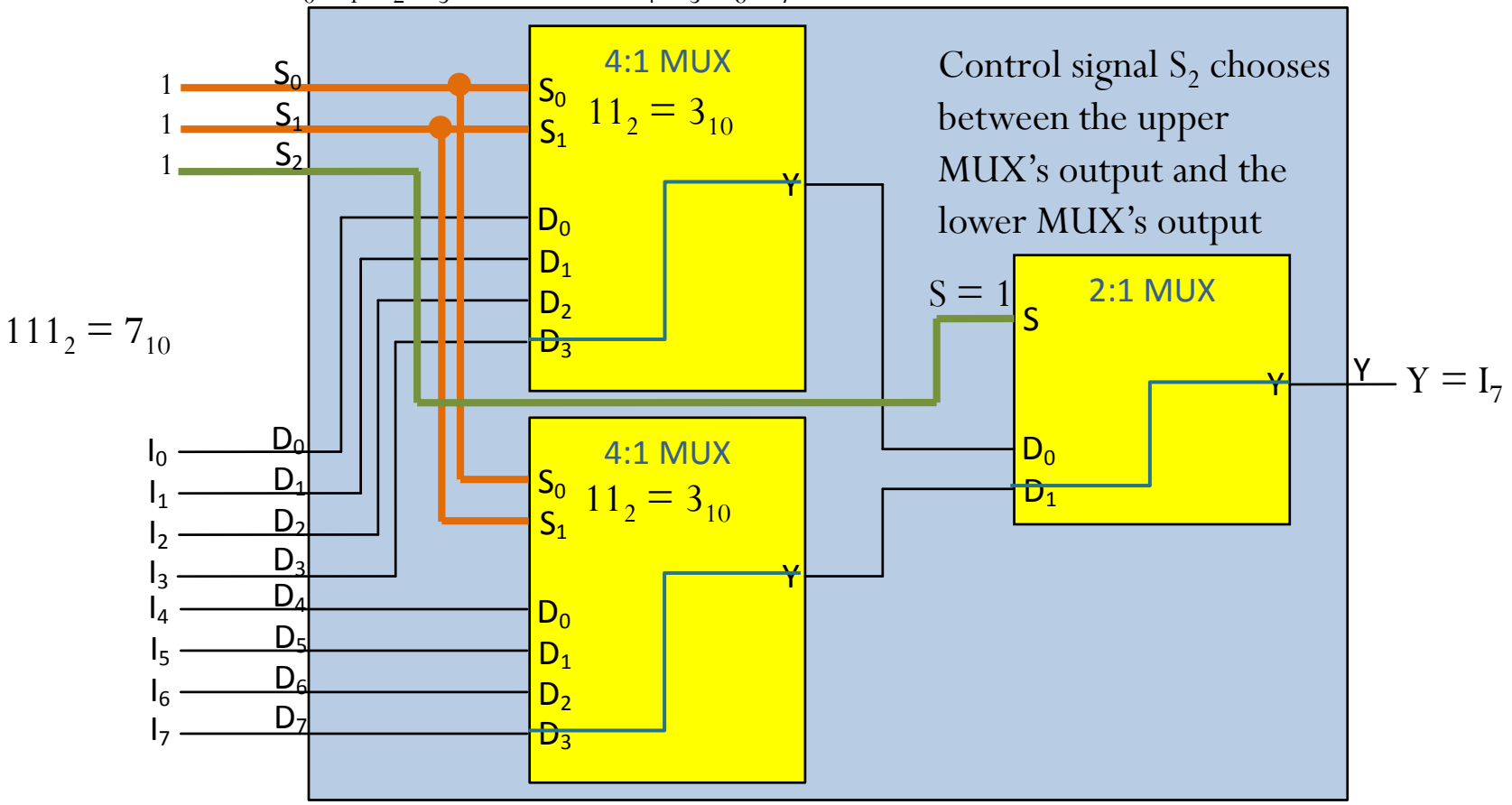
Multiplexer Expansion

Control signals S_0, S_1 simultaneously choose one of I_0, I_1, I_2, I_3 and one of I_4, I_5, I_6, I_7



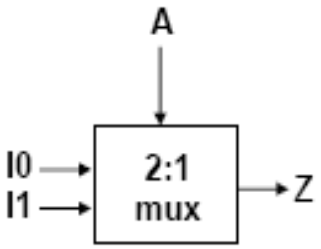
Multiplexer Expansion

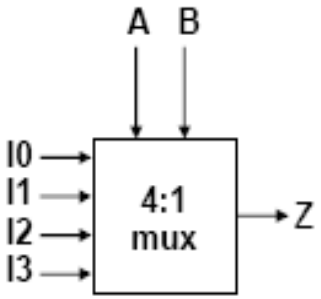
Control signals S_0, S_1 simultaneously choose one of I_0, I_1, I_2, I_3 and one of I_4, I_5, I_6, I_7

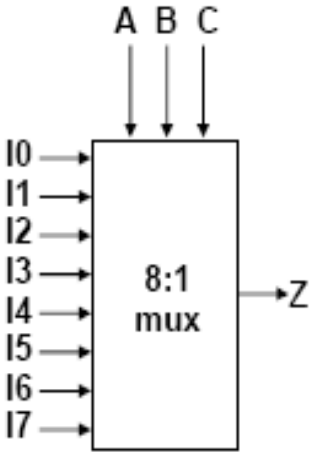


MUX and Minterms

- 2:1 mux:Z = A'I0 + AI1
- 4:1 mux:Z = A'B'I0 + A'BI1 + AB'I2 + ABI3
- 8:1 mux:Z = A'B'C'I0 + A'B'CI1 + A'BC'I2 + A'BCI3
  + AB'C'I4 + AB'CI5 + ABC'I6 + ABCI7



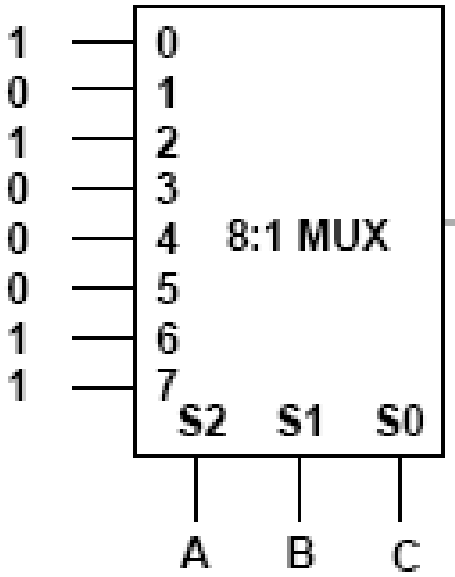




Mathematically, we may say that the output of the MUX is the weighted sum of all minterms (generated from the control variables) where the weights are the data inputs.

MUX as a Logic Function Generator

2^n:1 MUX can be used to implement any function of *n* variables.



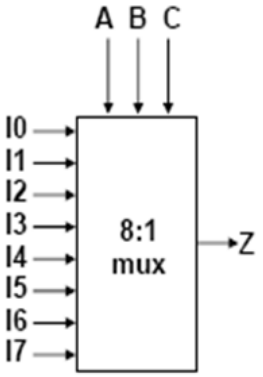
Note that no extra gate is needed!

Example:

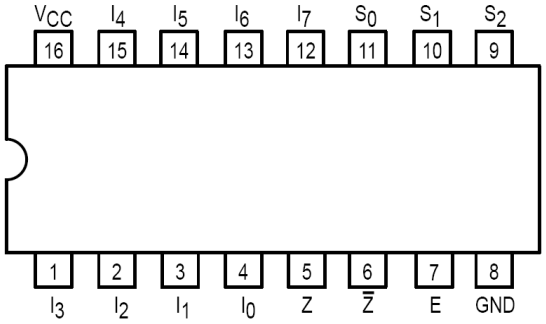
$$F = \sum_{A,B,C}(0,2,6,7)$$

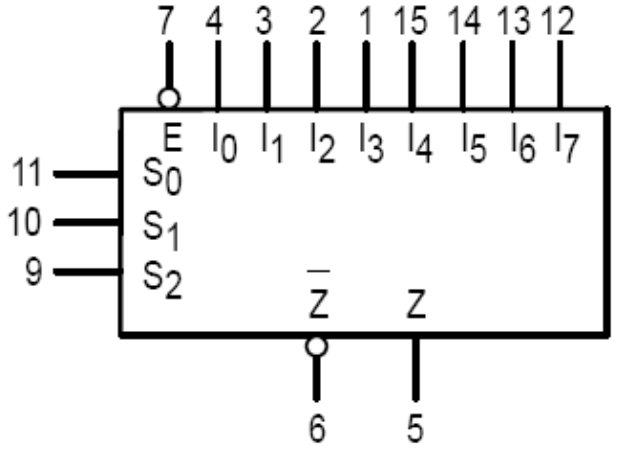
Explanation:

$$Z = \overline{A}\cdot\overline{B}\cdot\overline{C}\cdot I_0 + \overline{A}\cdot\overline{B}\cdot C\cdot I_1 + \overline{A}\cdot B\cdot\overline{C}\cdot I_2 + \overline{A}\cdot B\cdot C\cdot I_3$$
$$+ A\cdot\overline{B}\cdot\overline{C}\cdot I_4 + A\cdot\overline{B}\cdot C\cdot I_5 + A\cdot B\cdot\overline{C}\cdot I_6 + A\cdot B\cdot C\cdot I_7$$



74x151: 8:1 MUX

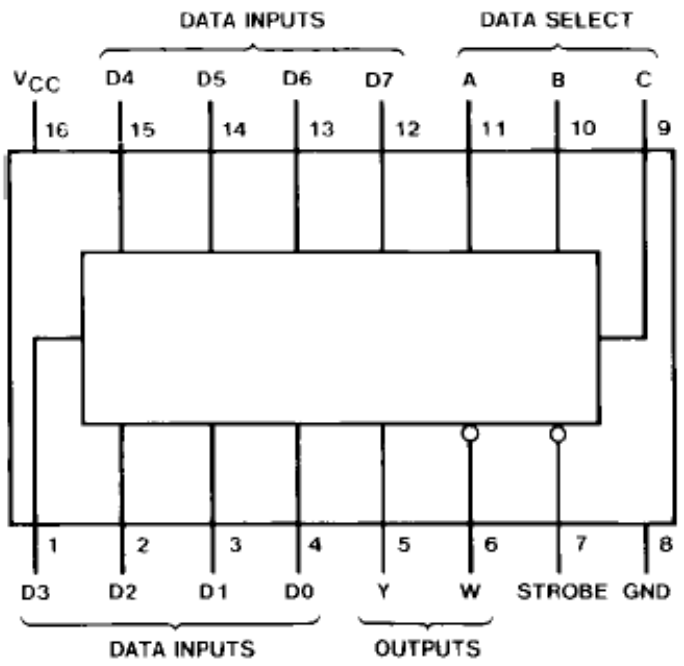




$\bar{E}$	S_2	S_1	S_0	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	$\bar{Z}$	Z
H	X	X	X	X	X	X	X	X	X	X	X	H	L
L	L	L	L	L	X	X	X	X	X	X	X	H	L
L	L	L	L	H	X	X	X	X	X	X	X	L	H
L	L	L	H	X	L	X	X	X	X	X	X	H	L
L	L	L	H	X	H	X	X	X	X	X	X	L	H
L	L	H	L	X	X	L	X	X	X	X	X	H	L
L	L	H	L	X	X	H	X	X	X	X	X	L	H
L	L	H	H	X	X	X	L	X	X	X	X	H	L
L	L	H	H	X	X	X	H	X	X	X	X	L	H
L	H	L	L	X	X	X	X	L	X	X	X	H	L
L	H	L	L	X	X	X	X	H	X	X	X	L	H
L	H	L	H	X	X	X	X	X	L	X	X	H	L
L	H	L	H	X	X	X	X	X	H	X	X	L	H
L	H	H	L	X	X	X	X	X	X	L	X	H	L
L	H	H	L	X	X	X	X	X	X	H	X	L	H
L	H	H	H	X	X	X	X	X	X	X	L	H	L
L	H	H	H	X	X	X	X	X	X	X	H	L	H

74x151: 8:1 MUX

Connection Diagram

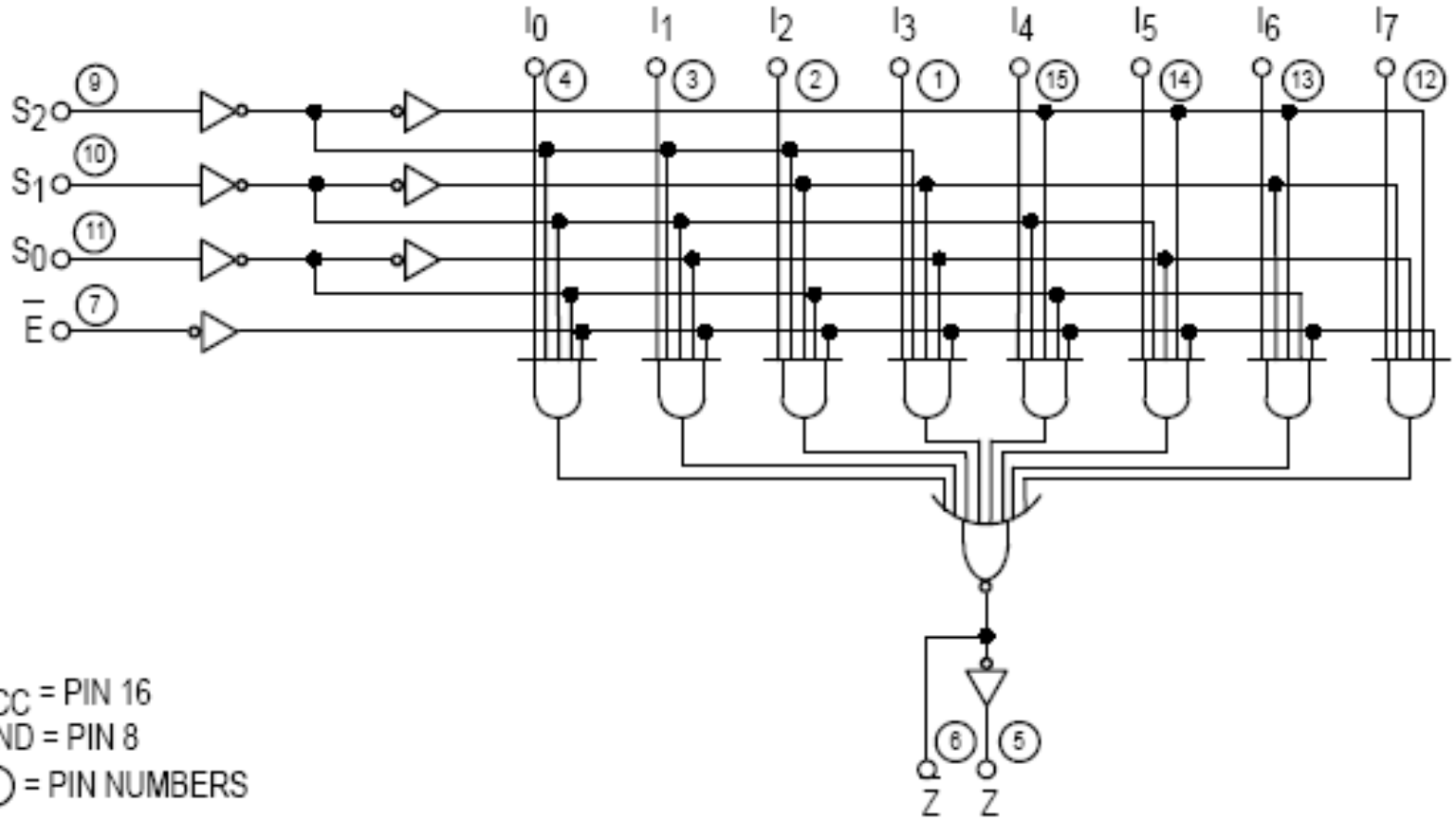


Truth Table

Inputs				Outputs	
Select			Strobe	Y	W
C	B	A	S		
X	X	X	H	L	H
L	L	L	L	D0	$\overline{D0}$
L	L	H	L	D1	$\overline{D1}$
L	H	L	L	D2	$\overline{D2}$
L	H	H	L	D3	$\overline{D3}$
H	L	L	L	D4	$\overline{D4}$
H	L	H	L	D5	$\overline{D5}$
H	H	L	L	D6	$\overline{D6}$
H	H	H	L	D7	$\overline{D7}$

H = HIGH Level
L = LOW Level
X = Don't Care
D0, D1...D7 = the level of the respective D input

74x151: Logic Diagram



74x157 Quad 2-Input MUX

- Four separate 2-input multiplexer.
- Each of the four multiplexers shares a common data-select line and a common Enable.