CS 221
Logic Design

Fall 2021

By Wessam El-Behaidy & Salwa Osama

Midterm Exam

- Midterm Exam (based on announced schedule)

Day: Wed. 24 Nov.

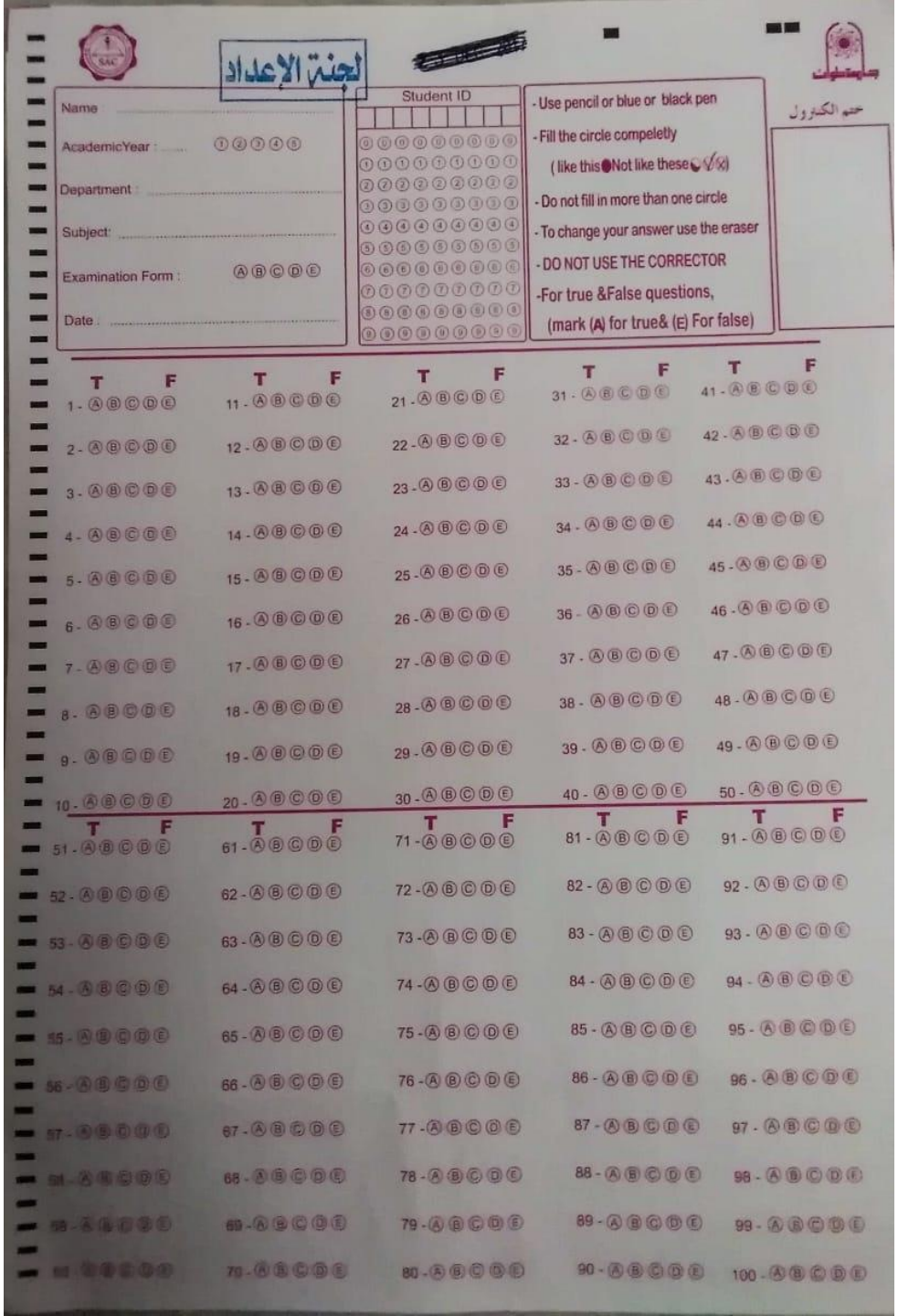
Time: from 11 for Group1 (800 students)

from 12 for Group2 (395 students)

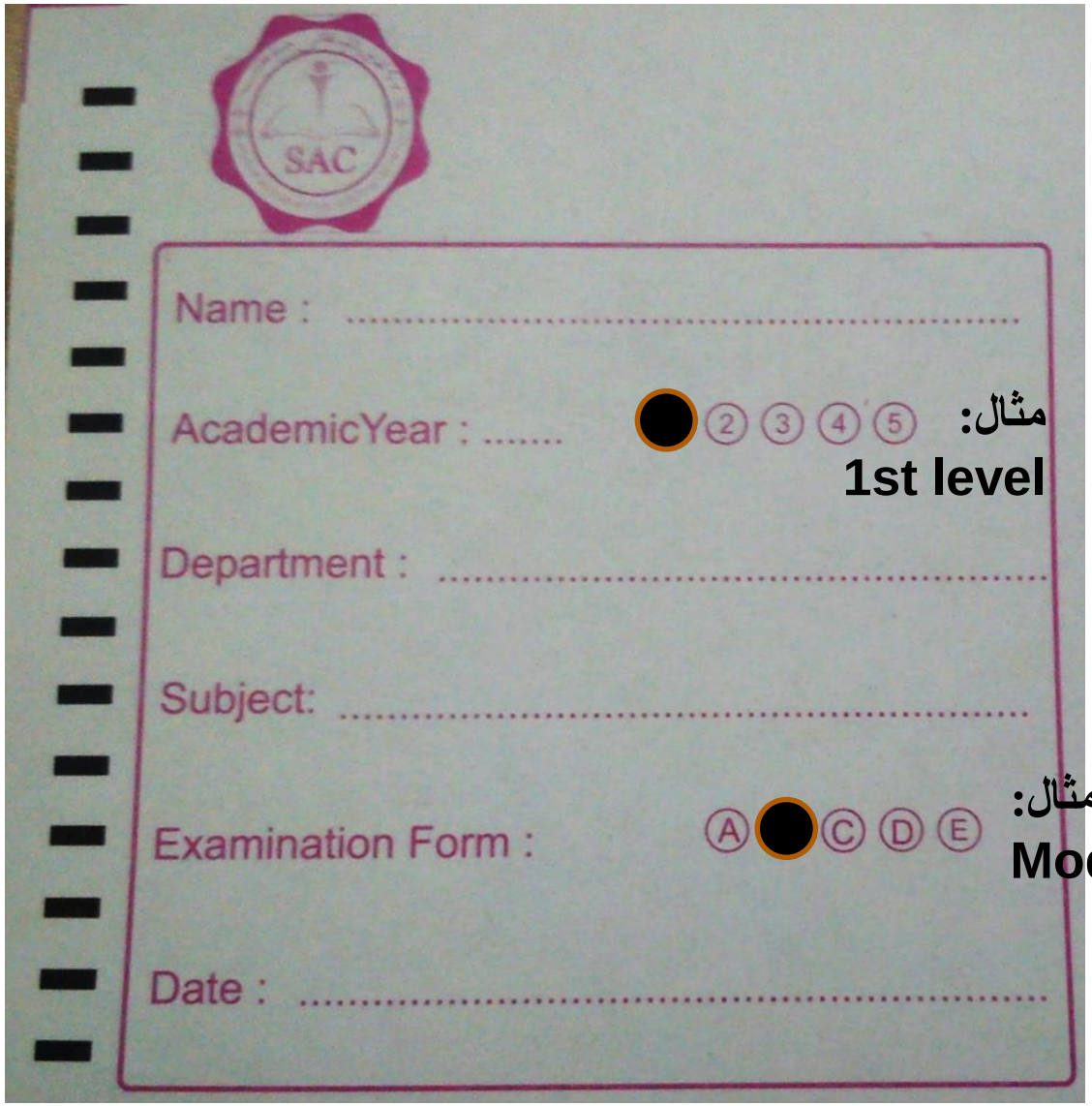
تنبيهات هامة

- الحضور للكلية قبل الموعد بنصف ساعة على الأقل
- يتواجد معك (قلم جاف- قلم رصاص – استيكة)
- الاجابة فى ورقة الاجابات الالكترونية بالقلم الرصاص وتحبر فى النهاية
- كتابة البيانات على ورقة الاسئلة وورقة الاجابة
- تسليم كلا من ورقة الاسئلة وورقة الاجابة (**عدم تسليم الورقتين يلغى امتحانك**)
- غير مسموح باستخدام الالة الحاسبة داخل الامتحان

الشكل العام لورقة التصحيح الألكترونى

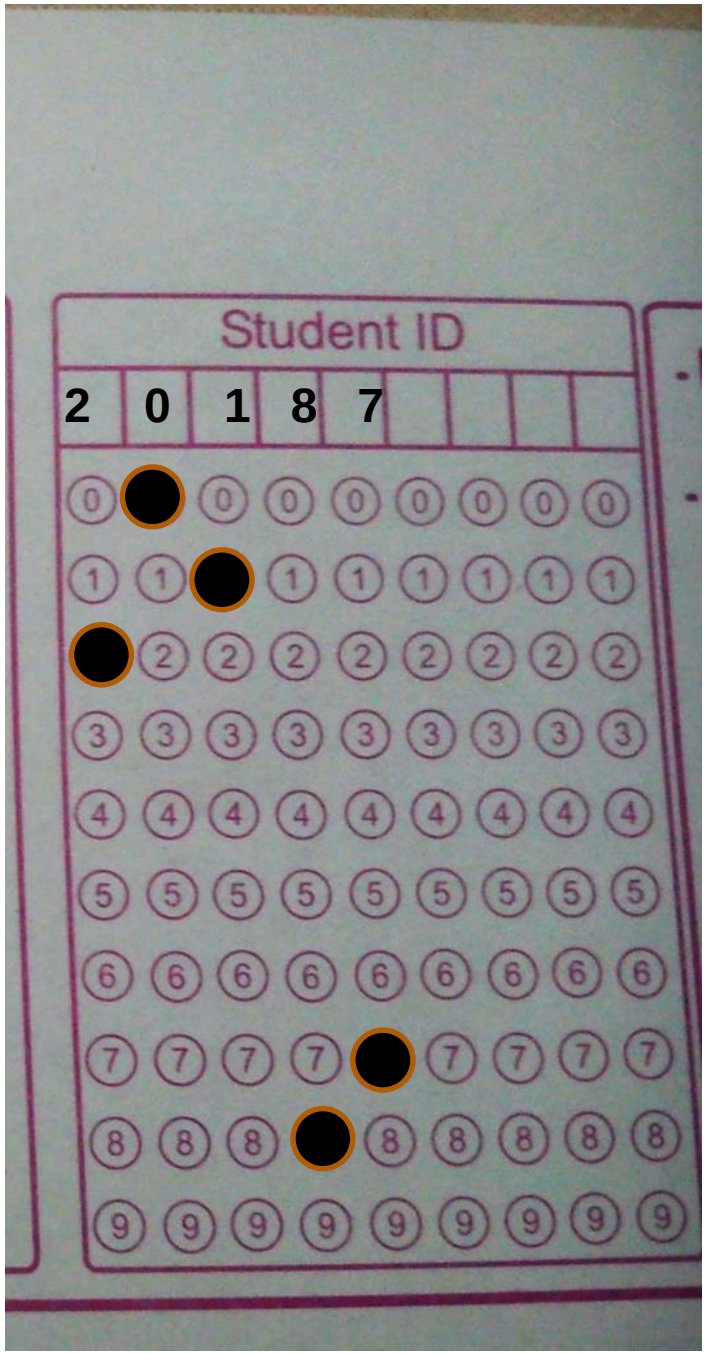


البيانات الأساسية



مثال:
1st level

مثال:
Model B

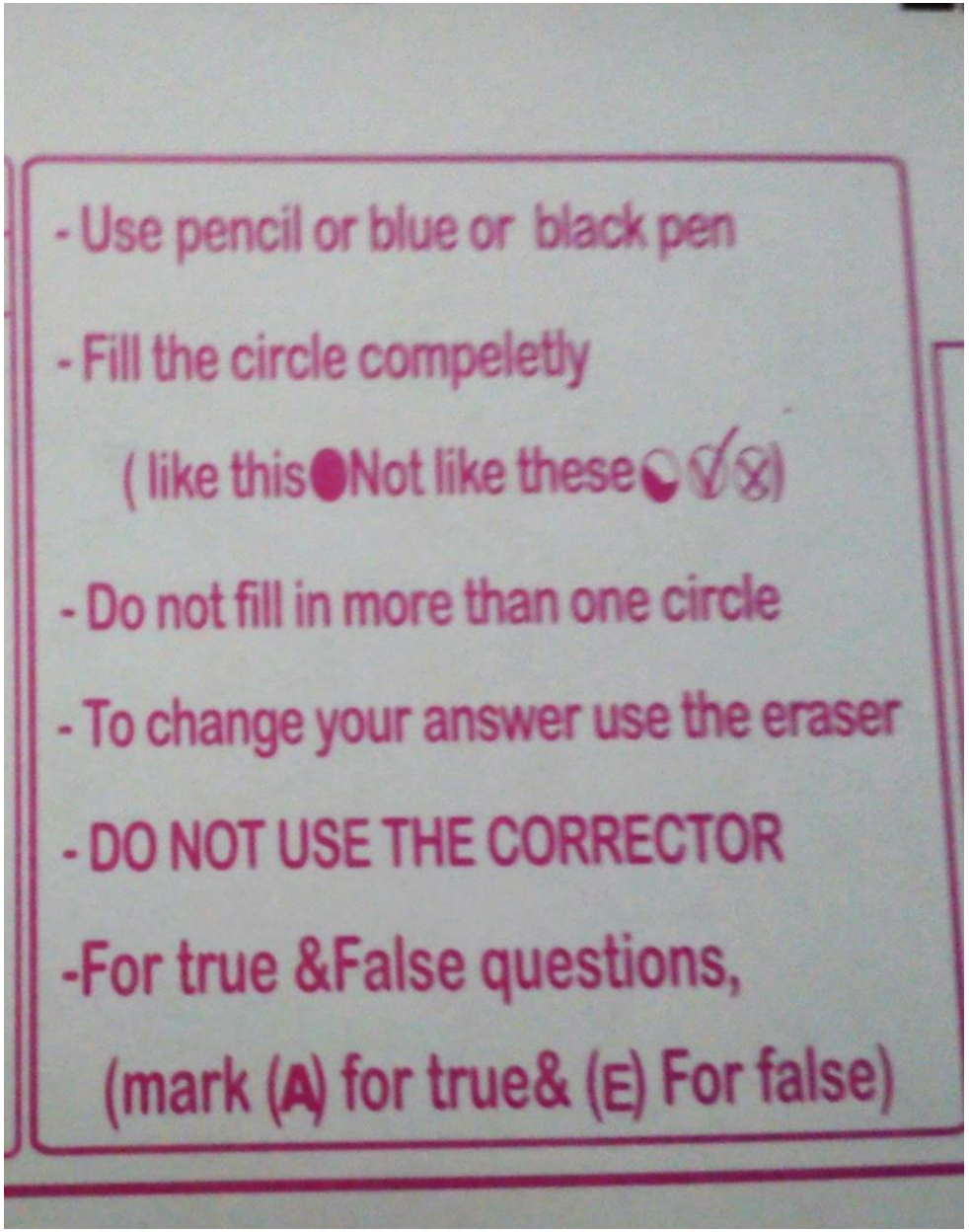


رقم الطالب

مثال:

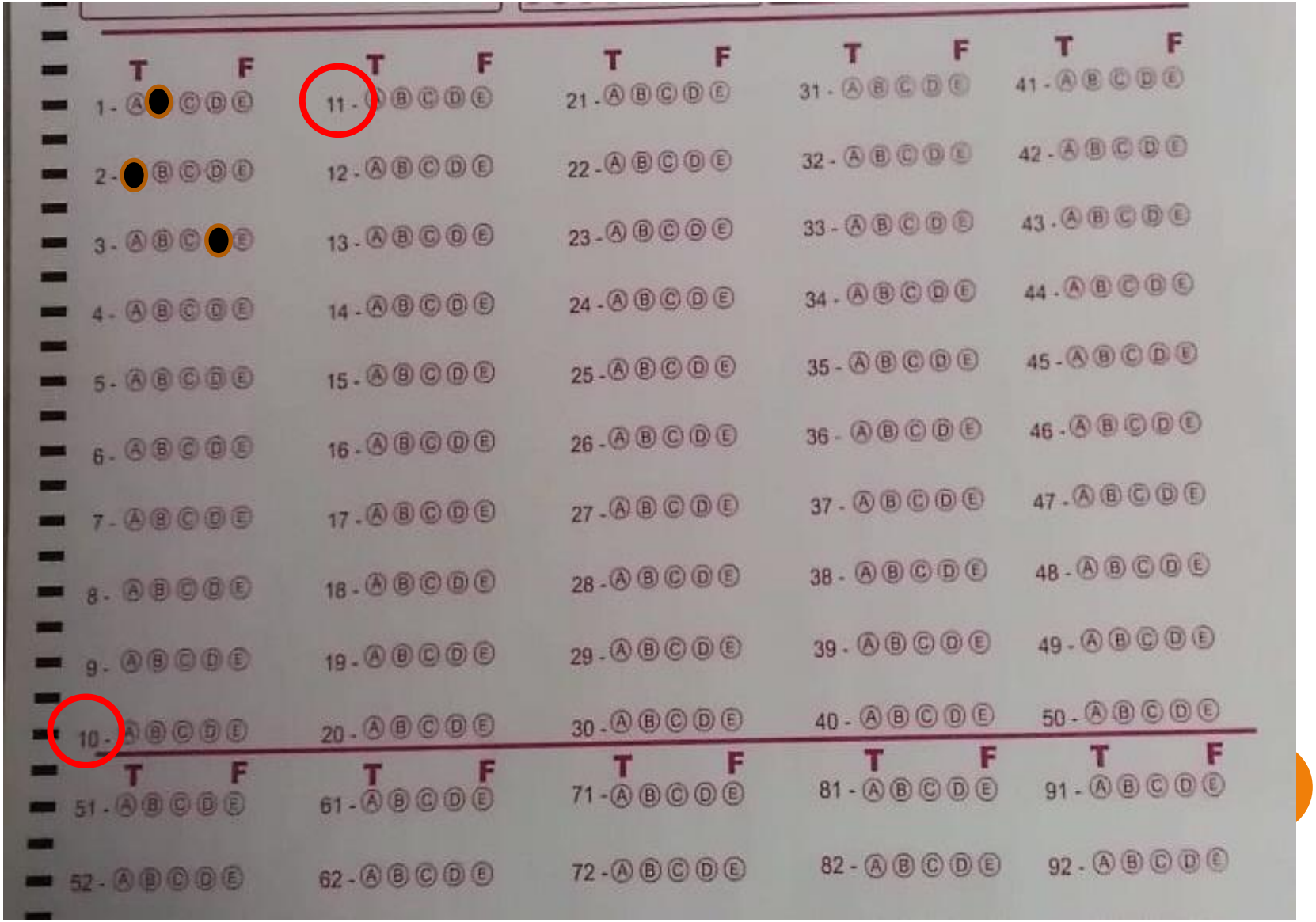
رقم الجلوس : 20187

تعليمات هامة

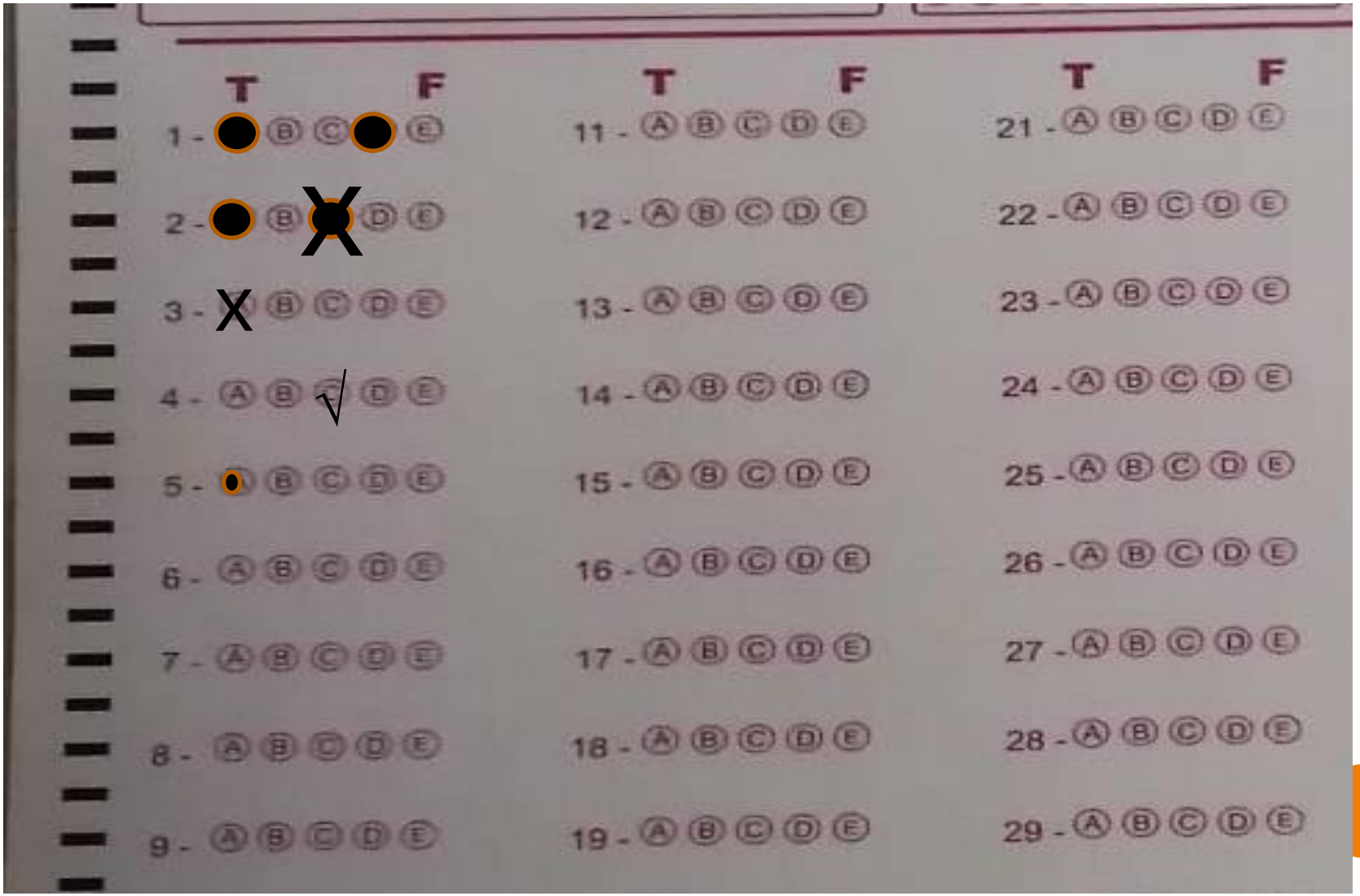


مثال لإجابات مظللة بشكل صحيح

مثال: اجابة السؤال الأول B



مثال لإجابات مظللة بشكل خطأ



INSTRUCTIONS OF THE EXAM

Faculty of Computers and Artificial Intelligence
CS 221: Logic Design
Fall 2021- 2022 Time: 30 minutes
Dr. Wessam El-Behaidy & Dr. Salwa Osama

MIDTERM EXAM
(Group2_Model A)

Helwan University

Please read the following instructions carefully:

- Remember to **fill in your Name and Student ID**. Only sheets with both a name and ID will be graded.
- **Deliver both external answer sheet and the question paper. If not, your paper will not graded**
- **No Calculator** is permitted
- This is a **closed-book exam**. You are forbidden from accessing any external material, notes, electronic material, or online resources.

Name:..... **ID:** .....

- Attempt to **answer ALL questions** in this exam.

Multiple choice Questions (Circle **only ONE answer per question** on the external answer sheet. If more than one answer is selected, the question will NOT be graded.) [20 marks]

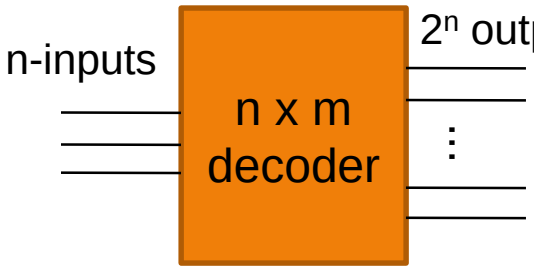
Combinational circuits

Lecture 6

REMEMBER

- This chapter includes the most important standard combinational circuits:
  - Adders, Subtractors, Decoders, Encoders, Multiplexers and Demultiplexer.
- We will know their internal design and the functionality of each.
- But, remember **our aim is to know how to think to design a circuit**

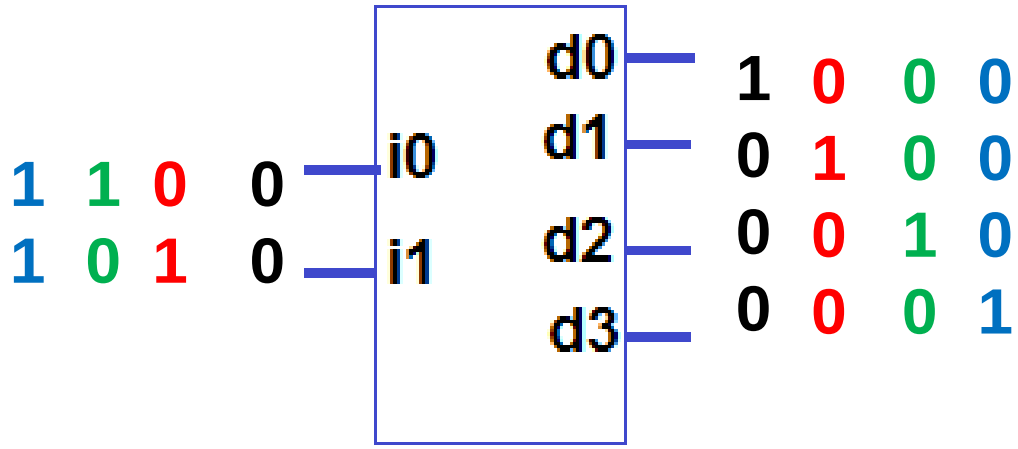
DECODERS



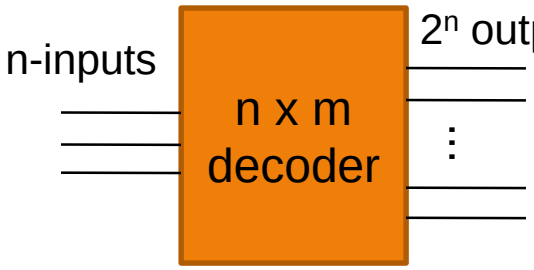
- A *Decoder*:
  - Is a popular combinational logic building block
  - It converts input binary number to one high output
- 2-input decoder

➔ Has four possible input binary numbers

➔ So, it has four outputs, one for each possible input binary number



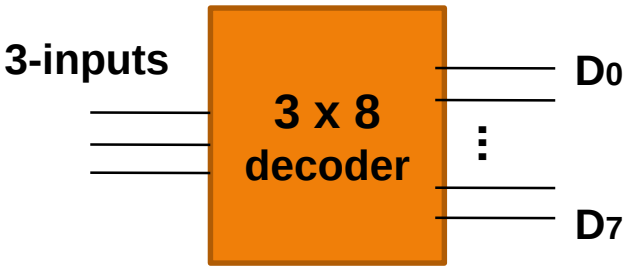
DECODERS



- Decoders are called *n-to-m* line decoders, where $m <= 2^n$.
- A particular application of this decoder is:

  Binary-to-octal conversion;
  - inputs : binary representation and
  - outputs : its correspondence in octal representation.

3-TO-8 LINE DECODER



- 3 input variables =? outputs

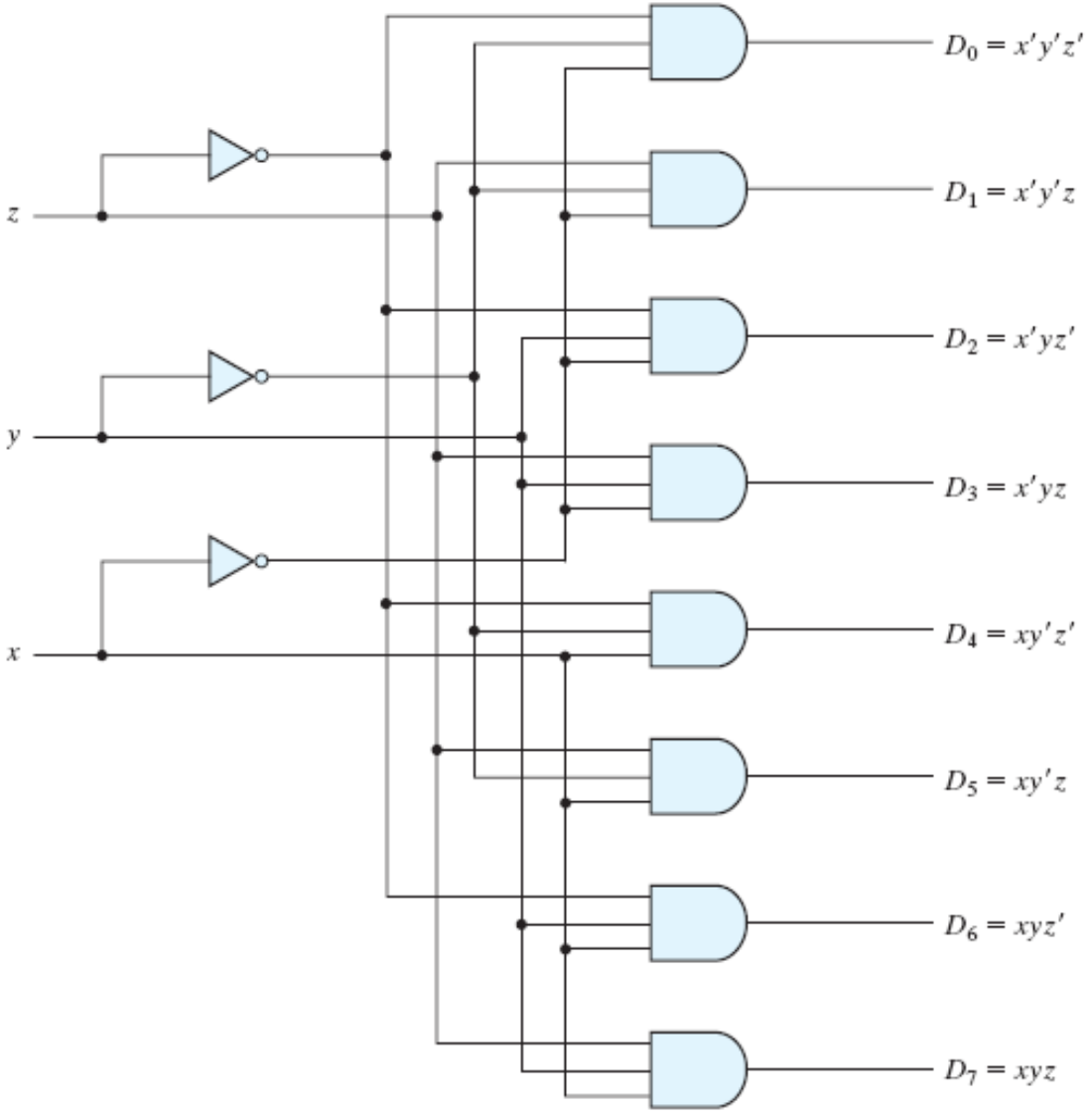
  ➔ 2^3=8 outputs

Table 4.6
Truth Table of a Three-to-Eight-Line Decoder

Inputs			Outputs							
x	y	z	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_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

D0= x'y'z', D1=x'y'z, D2=x'yz', D3=x'yz
D4= xy'z' D5=xy'z D6= xyz' D7=xyz

DECODERS

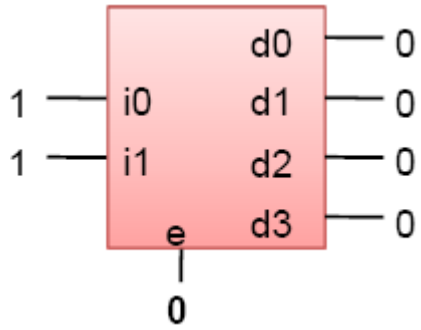


The output whose value is equal to 1 represents ***the minterm*** equivalent of the binary number currently available in the input lines

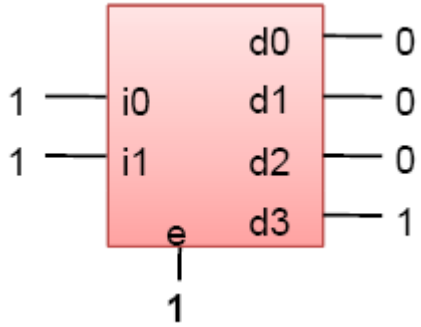
Decoder with enable input

Decoder with enable E

– if E=0, Outputs all 0



– if E=1, Regular behavior



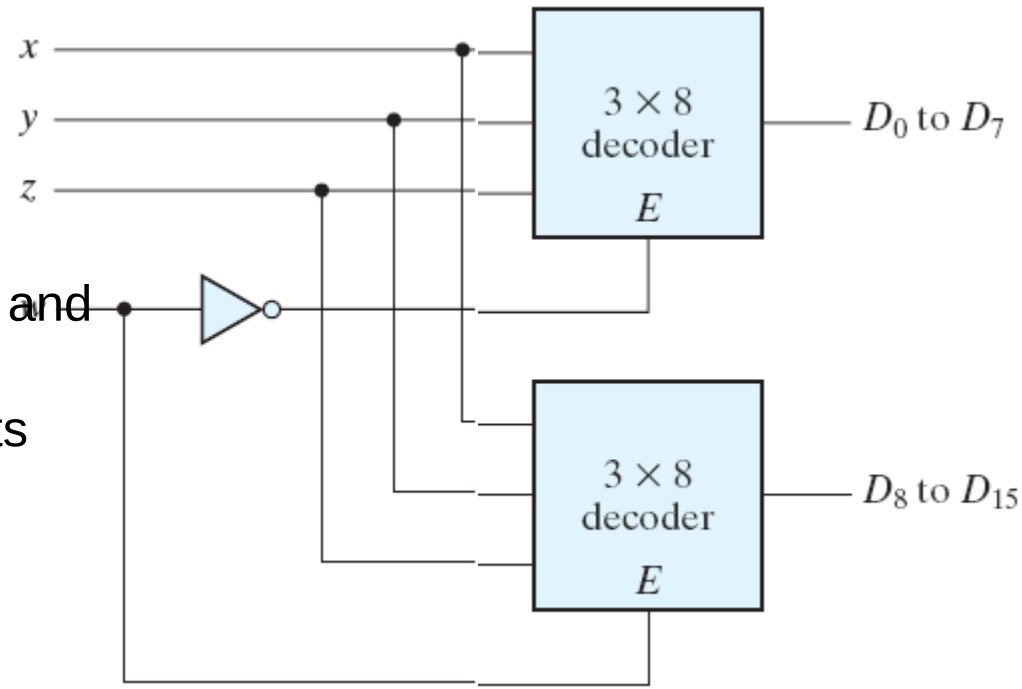
E	i0	i1	d0	d1	d2	d3
0	x	x	0	0	0	0
1	0	0	1	0	0	0
1	0	1		1		
1	1	0			1	
1	1	1				1

How to use as a block

4-TO-16 LINE DECODER

The 4-to-16 line decoder with two 3-to-8 line decoder with enable inputs

- When w=0,
  - the top decoder is enabled and the other is disabled
  - The bottom decoder outputs all 0's and the top eight outputs generate minterms 0000 to 0111
- When w=1,
  - the bottom decoder is enabled and the other is disabled
  - The top decoder outputs all 0's and the bottom eight outputs generate minterms 1000 to 1111

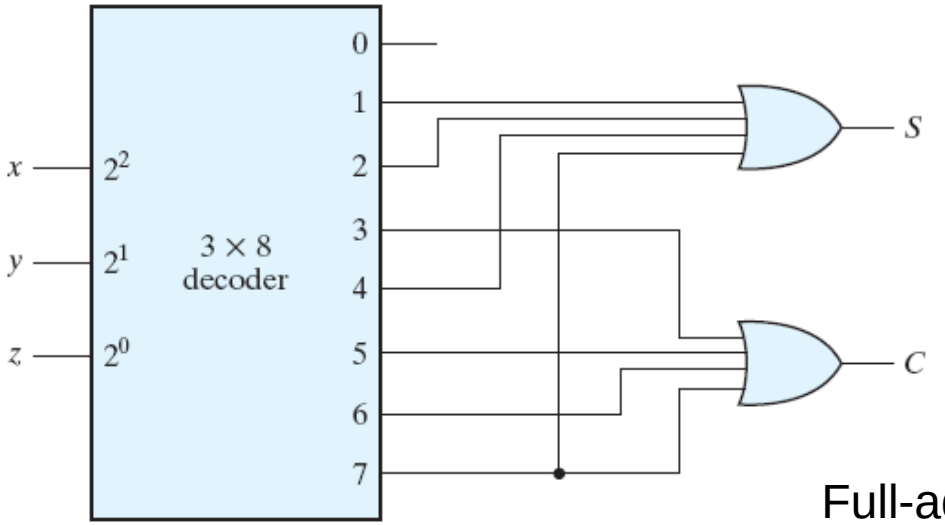


How to use as a block

BOOLEAN FUNCTION IMPLEMENTATION

- Since any Boolean function can be expressed in sum of minterms form:
  - ➔ A decoder with an external OR gate provides an implementation of the function
- Ex: Full-adder functions

  $S(x,y,z)= \Sigma(1,2,4,7)$ $C(x,y,z)=\Sigma(3,5,6,7)$



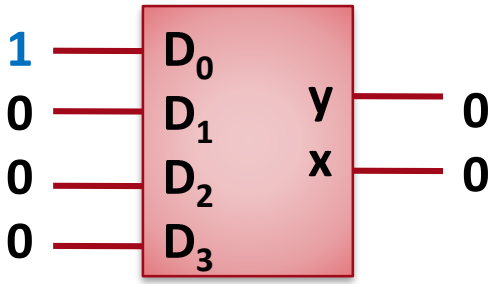
Full-adder with a decoder

NOTES

- A function with a long list of minterms requires an OR gate with large number of inputs
  - If the number of minterms $>2^n/2$, then F ' can be expressed with fewer minterms
  - In this case, NOR gate is used. Since the sum is complemented and the normal output is generated

ENCODERS

- An encoder
  - It is a digital circuit that performs the inverse operation of a decoder
  - It has 2^n inputs lines and n output lines
- An example of an encoder is:
  - Octal-to-binary conversion, it has:
    - 8 inputs : the octal code and
    - 3 outputs : the corresponding binary number
- It is assumed that only one input has a value of 1 at any given time



Octal to binary Encoder

Table 4.7
Truth Table of an Octal-to-Binary Encoder

Inputs								Outputs		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x	y	z
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

- It can be implemented with 3 OR gates

$$z = D_1 + D_3 + D_5 + D_7$$
$$y = D_2 + D_3 + D_6 + D_7$$
$$x = D_4 + D_5 + D_6 + D_7$$

- But, it has some limitations

Octal to binary Encoder: Limitations

- This encoder has some limitations:

1. It is assumed that only one input can be active at any given time,
   - If 2 inputs are active simultaneously, the output produces an undefined combination
   - **To resolve this ➔** higher priority with higher subscript
2. A output with all 0's is generated when all the input are 0;
   - But this output is the same as when D0 is equal to 1
   - **To resolve this ➔** one more output is used to indicate whether at least one input is equal to 1.

How to design?
Know internal ?

PRIORITY ENCODER

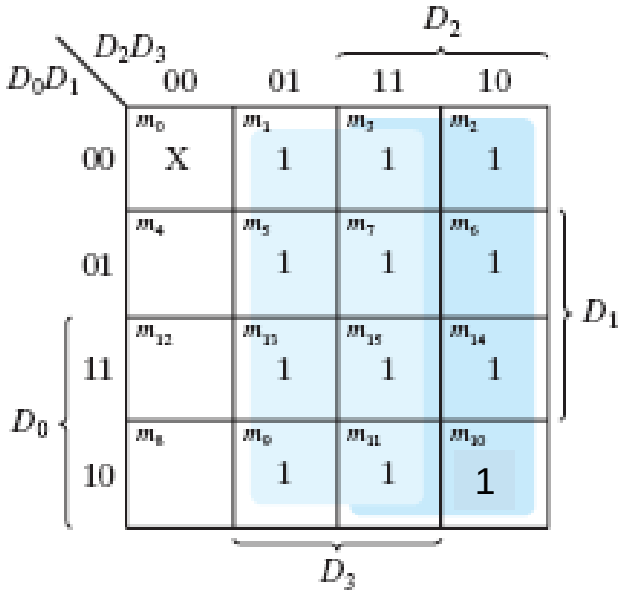
- A *priority encoder* is an encoder circuit that overcomes the limitations of octal-to-binary encoder previously implemented.

Table 4.8
Truth Table of a Priority Encoder

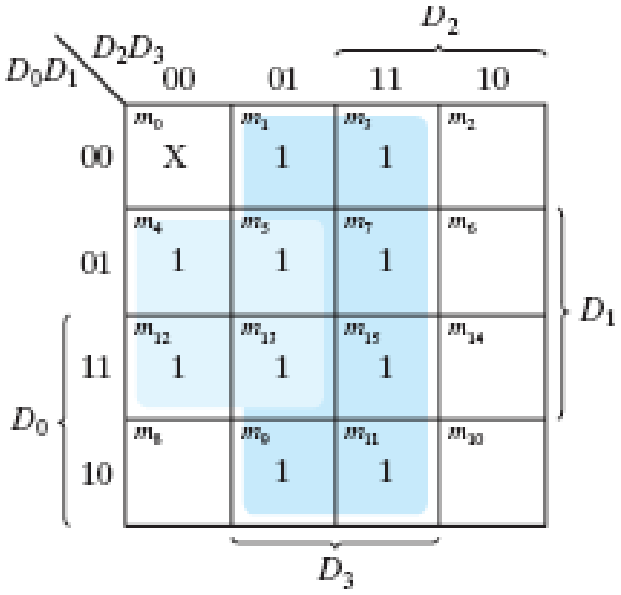
Inputs				Outputs		
D_0	D_1	D_2	D_3	x	y	V
0	0	0	0	X	X	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1

How to design?
Know internal ?

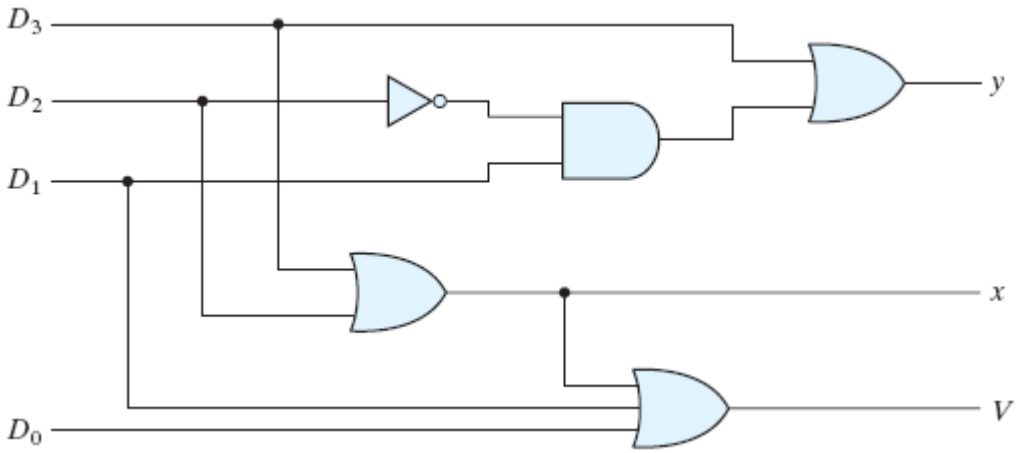
PRIORITY ENCODER(CONT.)



$x = D_2 + D_3$



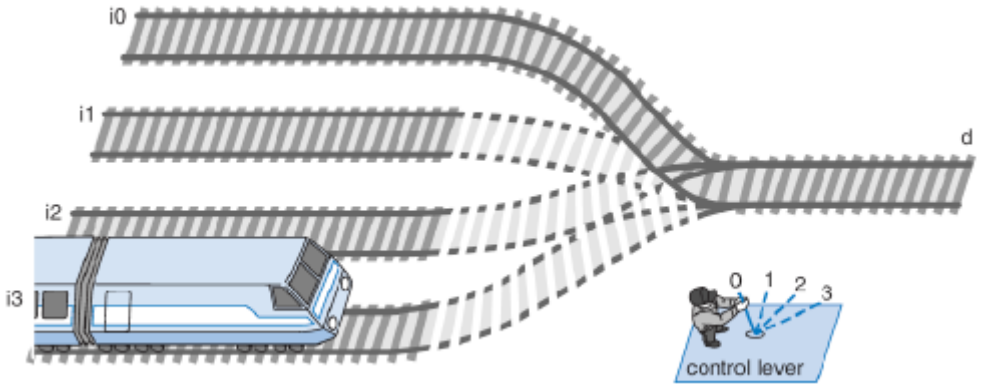
$y = D_3 + D_1D'_2$



MULTIPLEXERS

- A multiplexer (MUX) is a combinational circuit that
  - selects binary information from one of many input lines and
  - directs it to a single output line.

- It is also called a *Data selector*

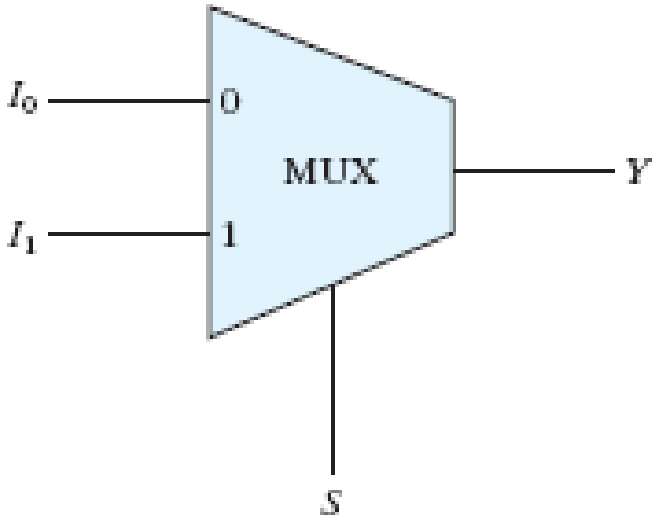


- There are:
  - 2^n input lines
  - *n* selection lines whose bit combinations determine which input is selected, and
  - Single output

TWO-TO-ONE MULTIPLEXER

It has:

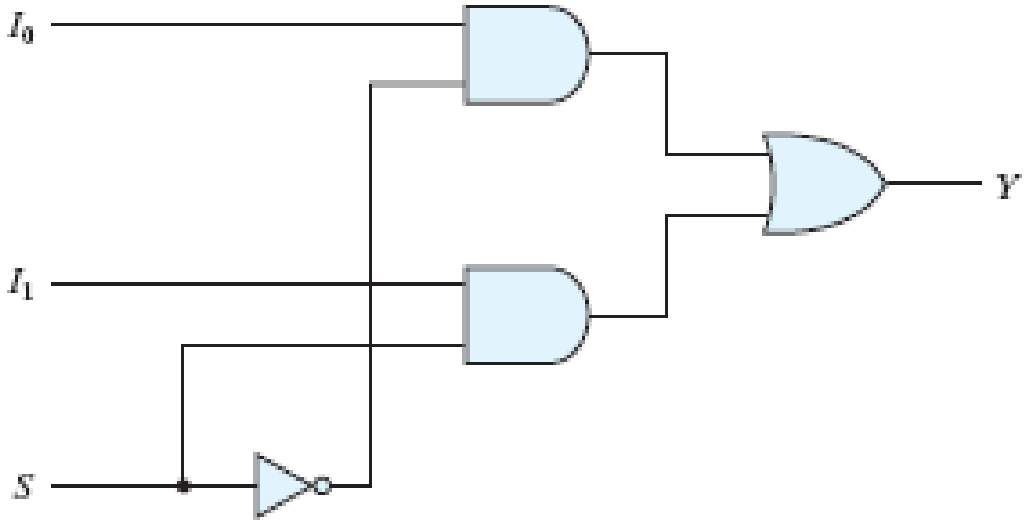
- 2 inputs, 1 selection and 1 output



(b) Block diagram

S	Y
0	I_0
1	I_1

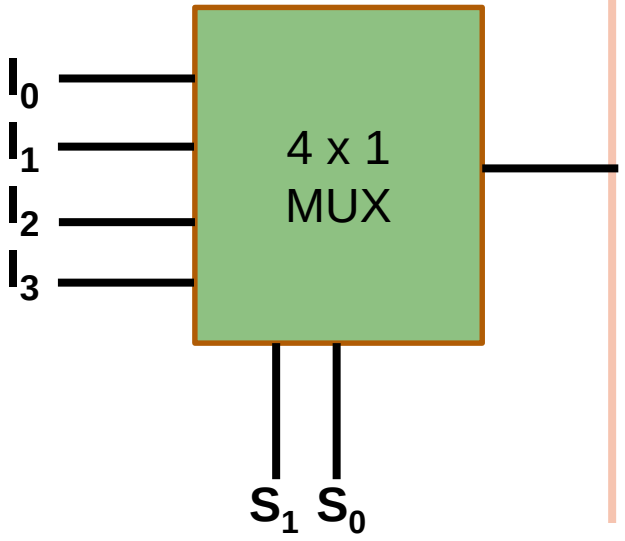
$Y = S'I_0 + SI_1$



(a) Logic diagram

How to design?
Know internal ?

FOUR-TO-ONE MULTIPLEXER

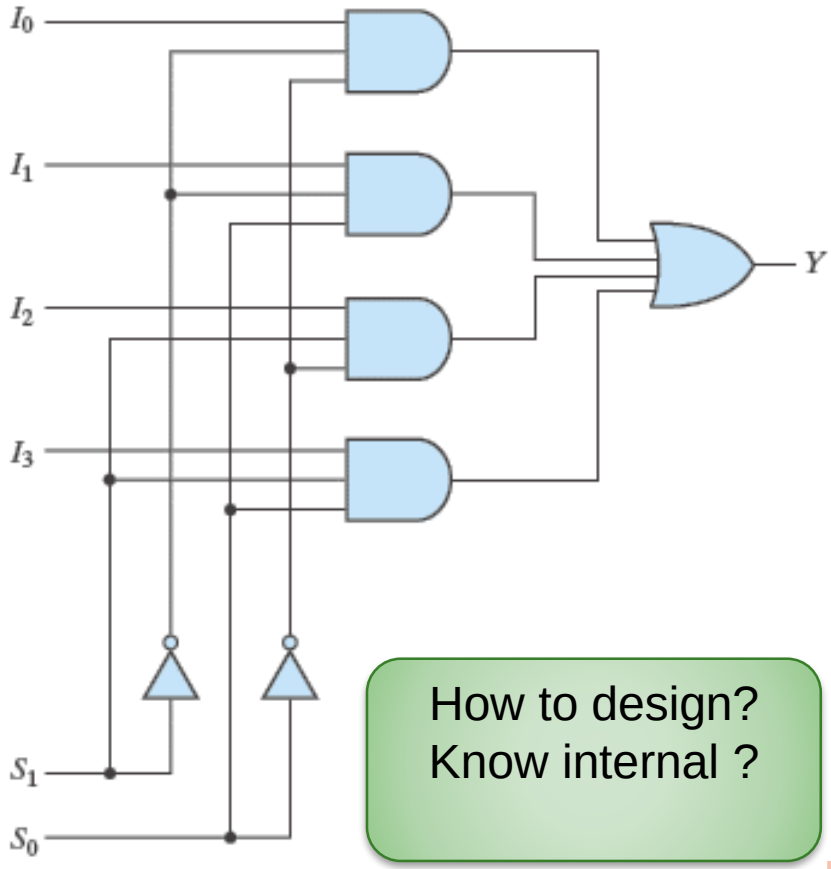


It has:

- 4 inputs, 2 selection and 1 output

S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

$Y = S_1'S_0'I_0 + S_1'S_0I_1 + S_1S_0'I_2 + S_1S_0I_3$

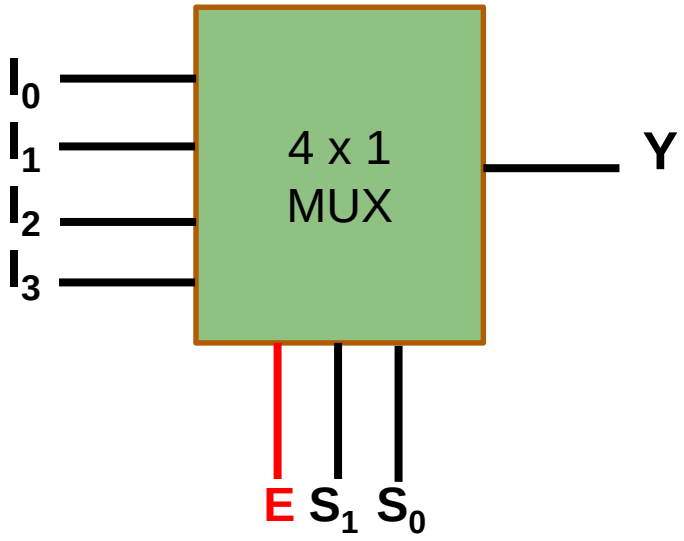


How to design?
Know internal ?

MULTIPLEXER WITH ENABLE INPUT

- As decoder, multiplexer may have an enable input:
  - When E=0, the outputs are disabled
  - When E=1, normal multiplexer

E	S1	S0	Y
0	x	X	All 0's
1	0	0	I_0
1	0	1	I_1
1	1	0	I_2
1	1	1	I_3



How to use as a block

QUADRUPLE 2-TO-1 LINE MULTIPLEXER

- The circuit has 4 MUXs,
  - Each MUX:
    - select one of two input lines: A_0 or B_0
    - Output Y_0
    - Selection line S selects one MUX from the 4 MUXs.
- It can be seen as a circuit that selects one of two 4-bit numbers
- The enable input E must be active for normal operation.

E	*S*	Output *Y*
1	X	all 0's
0	0	select *A*
0	1	select *B*

Function table

Quadruple 2-to-1 line multiplexer (Cont.)

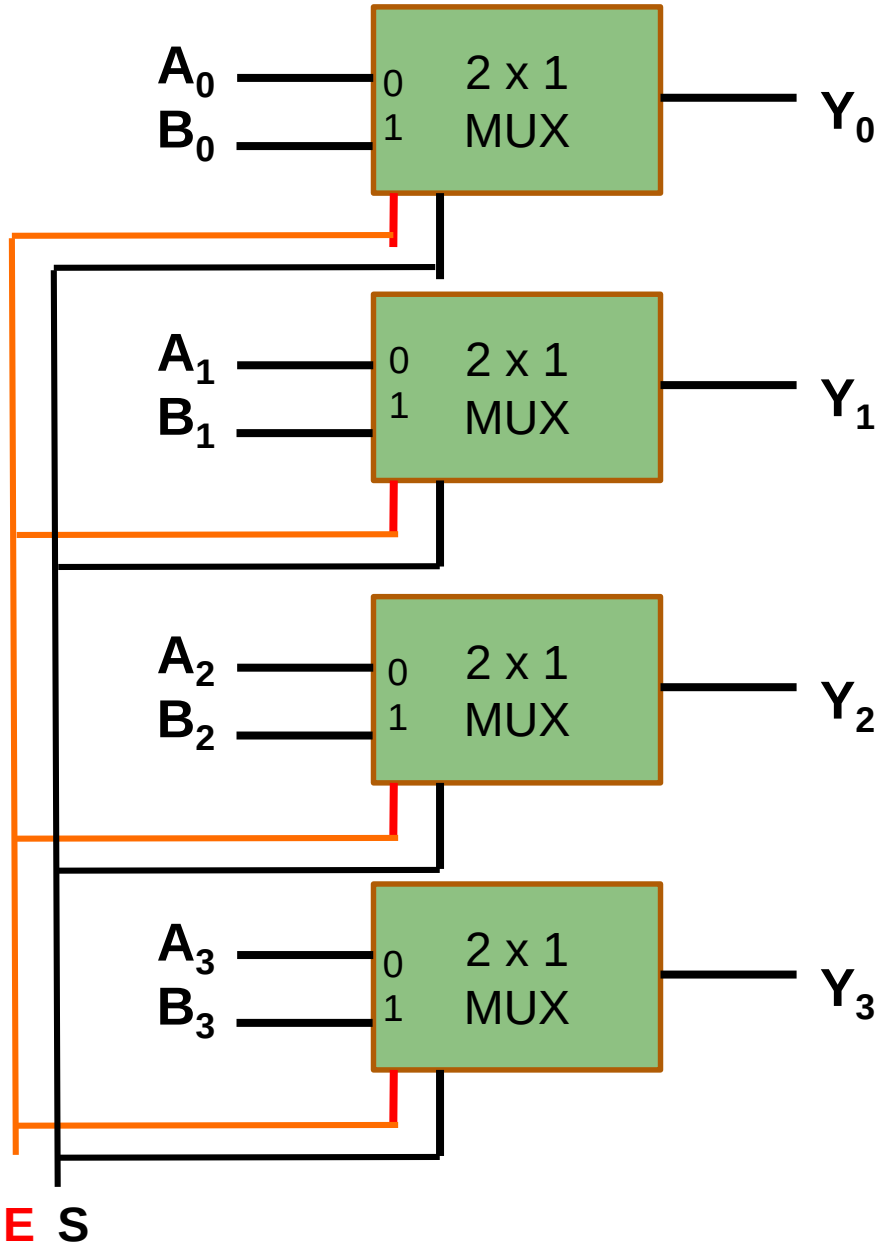
- The unit is enabled when E=0

If S=0, the four A inputs have a path to the four outputs

If S=1, the four B inputs have a path to the four outputs

- When E=1, the outputs are all 0's regardless the value of S

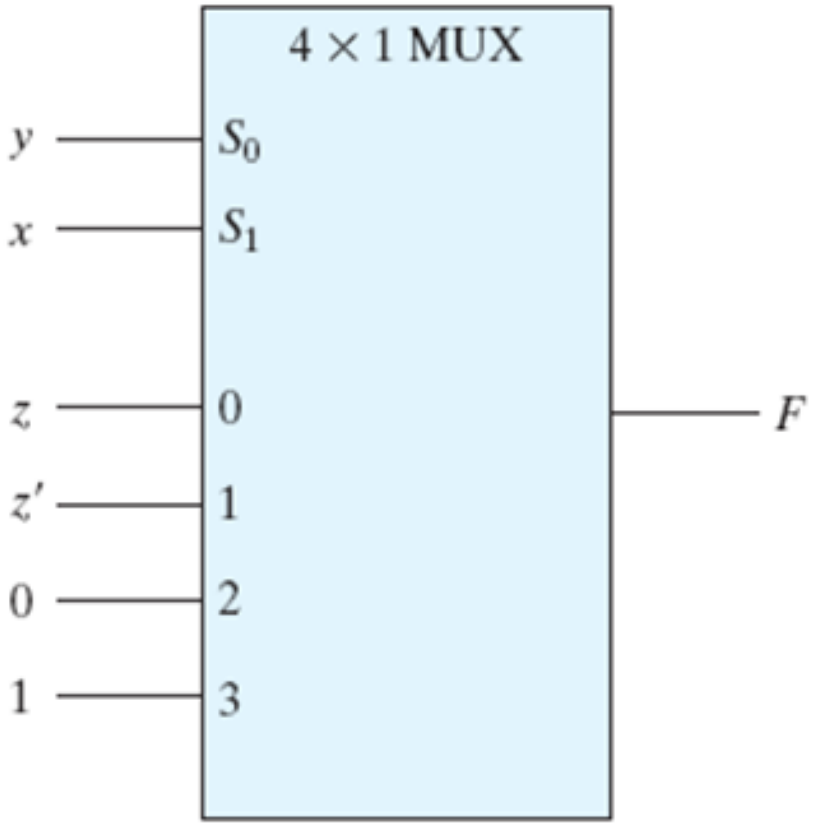
E	S	Output Y
1	X	all 0's
0	0	select A
0	1	select B



BOOLEAN FUNCTION IMPLEMENTATION

- *Example1:* $F(x,y,z)=\Sigma(1,2,6,7)$

x	y	z	F	
0	0	0	0	$F = z$
0	0	1	1	
0	1	0	1	$F = z'$
0	1	1	0	
1	0	0	0	$F = 0$
1	0	1	0	
1	1	0	1	$F = 1$
1	1	1	1	



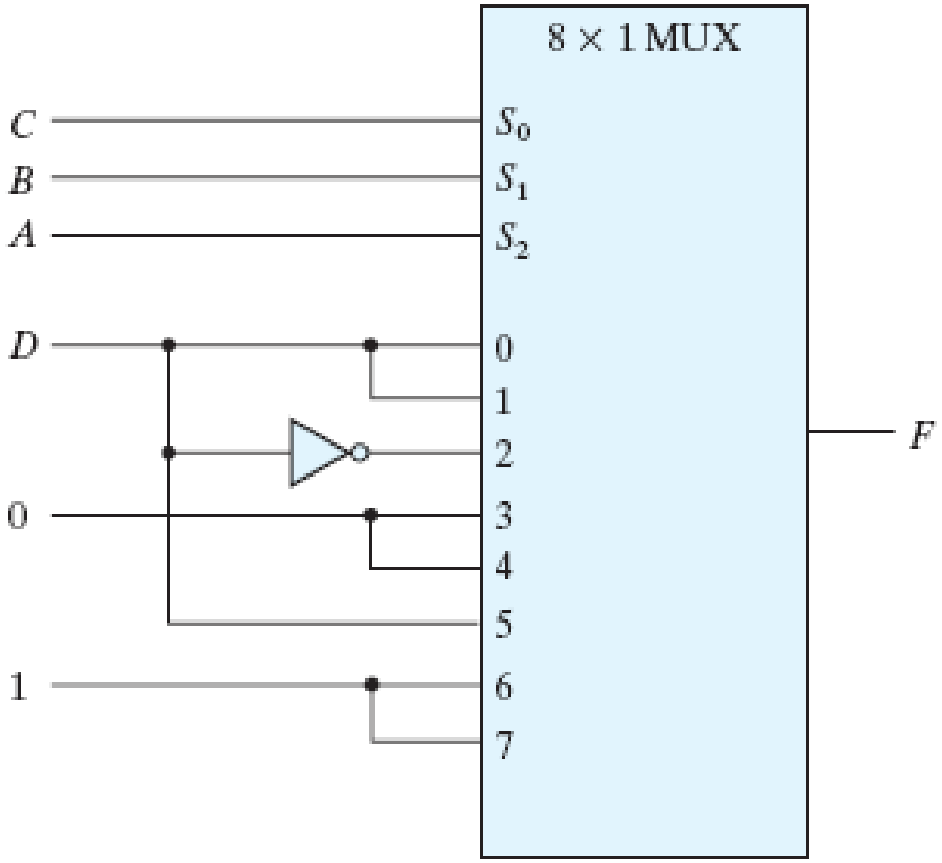
The first n-1 variables are applied to selection inputs and we evaluate the output as a function of the last variable

Boolean function implementation (Cont.)

How to use as a block

- *Example2: F(A, B, C, D)=Σ(1,3,4,11,12,13,14,15)*

A	B	C	D	F	
0	0	0	0	0	$F = D$
0	0	0	1	1	
0	0	1	0	0	$F = D$
0	0	1	1	1	
0	1	0	0	1	$F = D'$
0	1	0	1	0	
0	1	1	0	0	$F = 0$
0	1	1	1	0	
1	0	0	0	0	$F = 0$
1	0	0	1	0	
1	0	1	0	0	$F = D$
1	0	1	1	1	
1	1	0	0	1	$F = 1$
1	1	0	1	1	
1	1	1	0	1	$F = 1$
1	1	1	1	1	



MULTIPLEXER WITH A DECODER

- A 2^n-to-1 line multiplexer is constructed from:
  - An n-to-2^n decoder
  - AND gate for each 2^n input lines
  - An OR gate

try to solve it?

DEMULTIPLEXER

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

Fig: 4.19

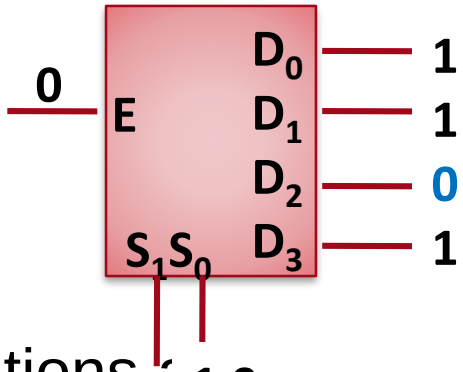
- Demultiplixer is

  A circuit that receives information from a single line and directs it to one of 2n output lines.

- The decoder with enable input can function as a demultiplexer.
- The decoder in fig. 4.9 can function

as 1-to-4 line demultiplexer, when:

  - E is taken as data input
  - A and B as selection inputs



- Because decoder and demultiplexer operations a obtained from the same circuit, a decoder with enable input is referred to as a *decoder-demultiplexer*

THANKS

We covered from:

Ch.4 (sec. 4.9→4.11)

Skip: three- state gates in sec. 4.11

Next lecture: Sequential circuits (latch & flip-flop)