ENGR 101
Engineering Technology

Dr. Kerese Manueli

School of Engineering and Computer Science
Victoria University of Wellington

Victoria
UNIVERSITY OF WELLINGTON
Te Whare Wānanga
o te Ūpoko o te Ika a Māui
CAPITAL CITY UNIVERSITY

Week 7 Lecture 12a

- Combinational circuit

- Assignment 2 – submit before midnight Monday
- Test 1 – Thursday 22 April (10:15 – 11:55am)

- Course web page:

  https://ecs.wgtn.ac.nz/Courses/XMUT101_2021T1/

- kerese@ecs.vuw.ac.nz

Designing Combinational Logic Circuits

Design Procedure:

1. Set up truth table
2. Write AND term for each case where the output is HI
3. Write the SOP expression for the output
4. Simplify the expression
5. Implement the circuit

Designing Combinational Logic Circuits

Example 1:

- Design a logic circuit that compares two 2-bit numbers, A and B. The circuit output will be high only when A > B.

2 bit numbers means A and B will have binary values of 00, 01, 10 and 11 (ie 0, 1, 2, & 3 in decimal)

1) Set up truth table

- Design a logic circuit that has 2-bit inputs *A*, and *B*, whose binary values are 00, 01, 10 and 11. In this case, we will use A0 to refer to the 1st bit and A1 refer to the 2nd bit. (same for B)

Inputs				Output
A1	**A0**	**B1**	**B0**	**G**
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

1) Set up truth table

The circuit output will be high only when A > B.

	Inputs				Output	
	A1	A0	B1	B0	G	
1st row →	0	0	0	0	0	Is A > B? → No
2nd row →	0	0	0	1	0	Is A > B? → No
3rd row →	0	0	1	0	0	Is A > B? → No
4th row →	0	0	1	1	0	Is A > B? → No
5th row →	0	1	0	0	1	Is A > B? → Yes
6th row →	0	1	0	1	0	Is A > B? → No
7th row →	0	1	1	0	0	Is A > B? → No
8th row →	0	1	1	1	0	Is A > B? → No
9th row →	1	0	0	0	1	Is A > B? → Yes
10th row →	1	0	0	1	1	Is A > B? → Yes
11th row →	1	0	1	0	0	Is A > B? → No
12th row →	1	0	1	1	0	Is A > B? → No
13th row →	1	1	0	0	1	Is A > B? → Yes
14th row →	1	1	0	1	1	Is A > B? → Yes
15th row →	1	1	1	0	1	Is A > B? → Yes
16th row →	1	1	1	1	0	Is A > B? → No

1) Set up truth table

2) Write AND term for each case where the output is HI

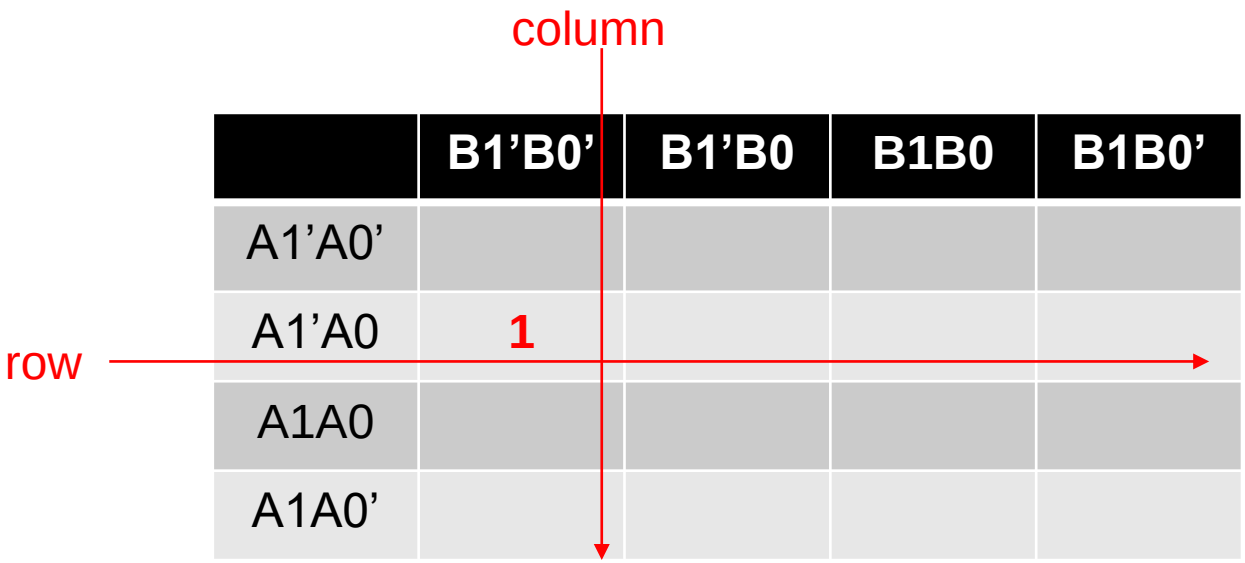
	Inputs				Output	
	A1	A0	B1	B0	G	
	0	0	0	0	0	
	0	0	0	1	0	
	0	0	1	0	0	
	0	0	1	1	0	
5th row →	0	1	0	0	1	A1'A0B1'B0'
	0	1	0	1	0	
	0	1	1	0	0	
	0	1	1	1	0	
9th row →	1	0	0	0	1	A1A0'B1'B0'
10th row →	1	0	0	1	1	A1A0'B1'B0
	1	0	1	0	0	
	1	0	1	1	0	
13th row →	1	1	0	0	1	A1A0B1'B0'
14th row →	1	1	0	1	1	A1A0B1'B0
15th row →	1	1	1	0	1	A1A0B1B0'
	1	1	1	1	0	

3) Write the SOP expression for the output

G = A1’A0B1’B0’ + A1A0’B1’B0’ +
A1A0’B1’B0 + A1A0B1’B0’ + A1A0B1’B0
+ A1A0B1B0’

4) Simplify the expression – using K-Map

G = A1’A0B1’B0’ + A1A0’B1’B0’ +
A1A0’B1’B0 + A1A0B1’B0’ + A1A0B1’B0
+ A1A0B1B0’



4) Simplify the expression – using K-Map

G = A1’A0B1’B0’ + A1A0’B1’B0’ +
A1A0’B1’B0 + A1A0B1’B0’ + A1A0B1’B0
+ A1A0B1B0’

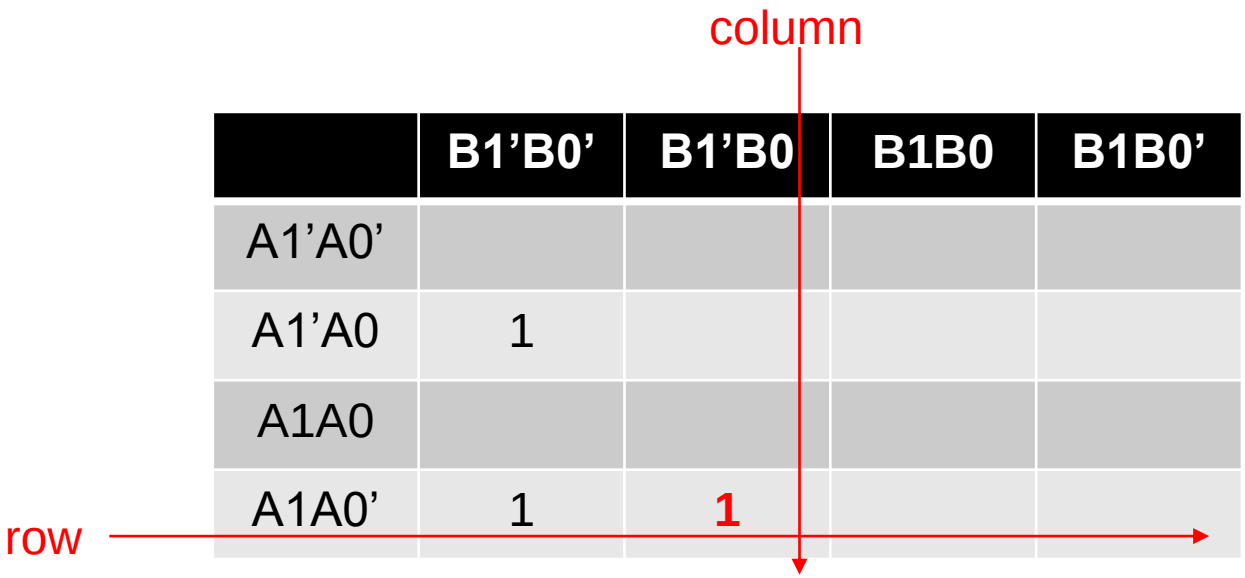
column

	B1’B0’	B1’B0	B1B0	B1B0’
A1’A0’				
A1’A0	1			
A1A0				
A1A0’	**1**			

row

4) Simplify the expression – using K-Map

G = A1’A0B1’B0’ + A1A0’B1’B0’ +
A1A0’B1’B0 + A1A0B1’B0’ + A1A0B1’B0
+ A1A0B1B0’



	B1’B0’	B1’B0	B1B0	B1B0’
A1’A0’				
A1’A0	1			
A1A0				
A1A0’	1	**1**		

4) Simplify the expression – using K-Map

G = A1’A0B1’B0’ + A1A0’B1’B0’ +
A1A0’B1’B0 + A1A0B1’B0’ + A1A0B1’B0
+ A1A0B1B0’

column

row

	B1’B0’	B1’B0	B1B0	B1B0’
A1’A0’				
A1’A0	1			
A1A0	**1**			
A1A0’	1	1		

4) Simplify the expression – using K-Map

G = A1’A0B1’B0’ + A1A0’B1’B0’ +
A1A0’B1’B0 + A1A0B1’B0’ + A1A0B1’B0
+ A1A0B1B0’

column

	B1’B0’	B1’B0	B1B0	B1B0’
A1’A0’				
A1’A0	1			
A1A0	1	**1**		
A1A0’	1	1		

row

4) Simplify the expression – using K-Map

G = A1’A0B1’B0’ + A1A0’B1’B0’ +
A1A0’B1’B0 + A1A0B1’B0’ + A1A0B1’B0
+ A1A0B1B0’

column

	B1’B0’	B1’B0	B1B0	B1B0’
A1’A0’				
A1’A0	1			
A1A0	1	1		**1**
A1A0’	1	1		

row

4) Simplify the expression – using K-Map

G = A1’A0B1’B0’ + A1A0’B1’B0’ +
A1A0’B1’B0 + A1A0B1’B0’ + A1A0B1’B0
+ A1A0B1B0’

	B1’B0’	B1’B0	B1B0	B1B0’
A1’A0’				
A1’A0	1			
A1A0	1	1		1
A1A0’	1	1		

4) Simplify the expression – using K-Map

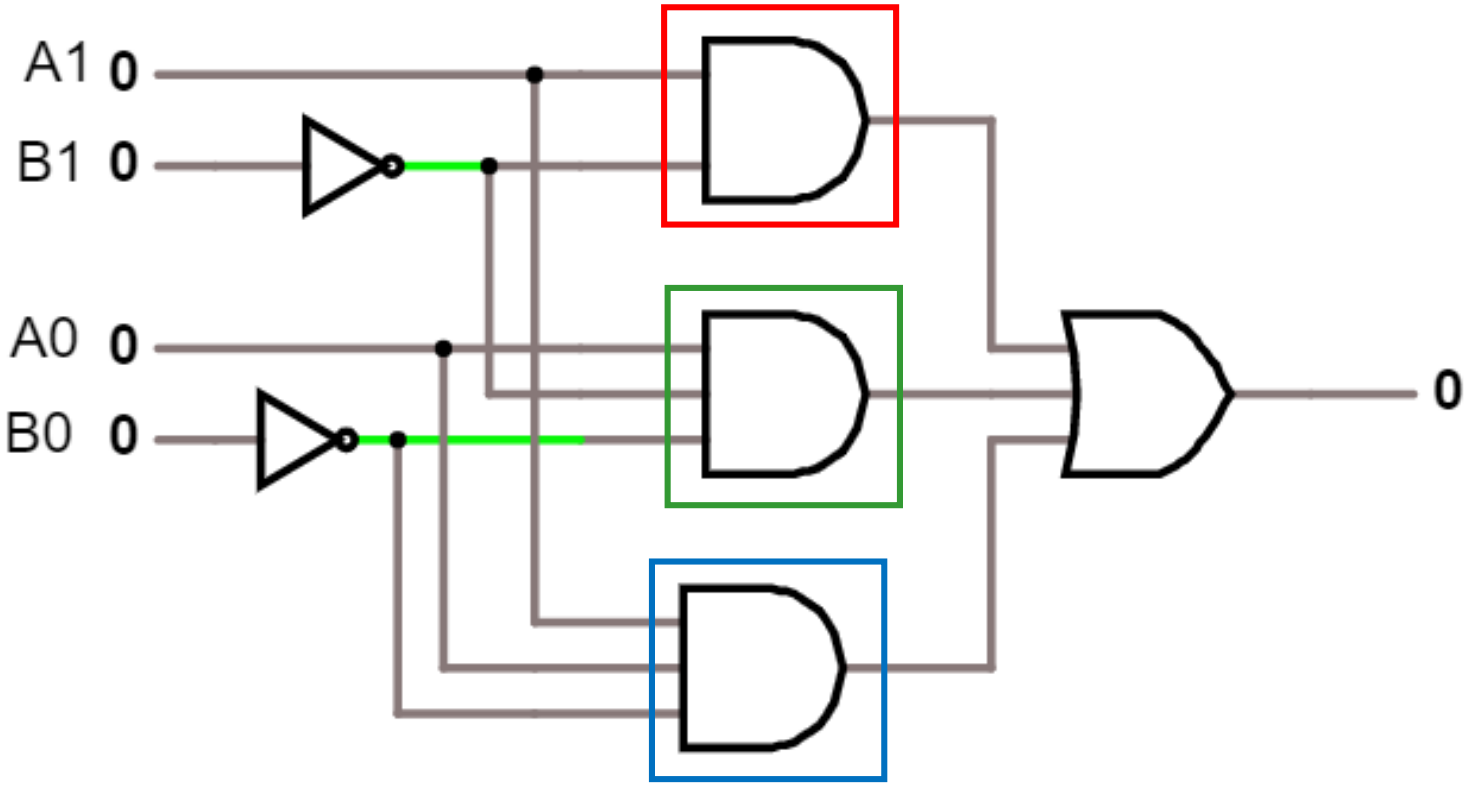
G = A1’A0B1’B0’ + A1A0’B1’B0’ +
A1A0’B1’B0 + A1A0B1’B0’ + A1A0B1’B0
+ A1A0B1B0’

	B1’B0’	B1’B0	B1B0	B1B0’
A1’A0’				
A1’A0	1			
A1A0	1	1		1
A1A0’	1	1		

Simplified expression is A1B1’ + A0B1’B0’ + A1A0B0’

5) Implement circuit:

Simplified expression is A1B1’ + A0B1’B0’ + A1A0B0’



Designing Combinational Logic Circuits

Exercise for you!!

- Design a logic circuit whose output is High only for even numbers between 0 – 15. (Assume 0 is not an even number)

Hint:

An even number is an integer that can be divided exactly by 2.

1) Set up truth table

- Design a logic circuit whose output is High for even numbers between 0 – 15. (Assume 0 is not an even number)

Inputs				Output
A	B	C	D	
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

1) Set up truth table

- Design a logic circuit whose output is High for even numbers between 0 – 15. (Assume 0 is not an even number)

	Inputs				Output	
	A	B	C	D		
1st row →	0	0	0	0	0	Even number? → No
2nd row →	0	0	0	1		Even number? → No
3rd row →	0	0	1	0	1	Even number → Yes
4th row →	0	0	1	1		Even number? → No
5th row →	0	1	0	0	1	Even number → Yes
6th row →	0	1	0	1		Even number? → No
7th row →	0	1	1	0	1	Even number → Yes
8th row →	0	1	1	1		Even number? → No
9th row →	1	0	0	0	1	Even number → Yes
10th row →	1	0	0	1		Even number? → No
11th row →	1	0	1	0	1	Even number → Yes
12th row →	1	0	1	1		Even number? → No
13th row →	1	1	0	0	1	Even number → Yes
14th row →	1	1	0	1		Even number? → No
15th row →	1	1	1	0	1	Even number → Yes
16th row →	1	1	1	1		Even number? → No

1) Set up truth table

2) Write AND term for each case where the output is HI

	Inputs				Output	
	A	B	C	D		
	0	0	0	0	0	
	0	0	0	1	0	
3rd row →	0	0	1	0	1	A'B'CD'
	0	0	1	1	0	
5th row →	0	1	0	0	1	A'BC'D'
	0	1	0	1	0	
7th row	0	1	1	0	1	A'BCD'
	0	1	1	1	0	
9th row	1	0	0	0	1	AB'C'D'
	1	0	0	1	0	
11th row	1	0	1	0	1	AB'CD'
	1	0	1	1	0	
13th row	1	1	0	0	1	ABC'D'
	1	1	0	1	0	
15th row	1	1	1	0	1	ABCD'
	1	1	1	1	0	

3) Write the SOP expression for the output

E = A’B’CD’ + A’BC’D’ + A’BCD’ + AB’C’D’ + AB’CD’ + ABC’D’ + ABCD’

4) Simplify the expression – using K-Map

E = A’B’CD’ + A’BC’D’ + A’BCD’ + AB’C’D’ + AB’CD’ + ABC’D’ + ABCD’

	C’D’	C’D	CD	CD’
A’B’				1
A’B				
AB				
AB’				

4) Simplify the expression – using K-Map

E = A’B’CD’ + A’BC’D’ + A’BCD’ + AB’C’D’ + AB’CD’ + ABC’D’ + ABCD’

	C’D’	C’D	CD	CD’
A’B’				1
A’B	1			
AB				
AB’				

4) Simplify the expression – using K-Map

E = A’B’CD’ + A’BC’D’ + A’BCD’ + AB’C’D’ + AB’CD’ + ABC’D’ + ABCD’

	C’D’	C’D	CD	CD’
A’B’				1
A’B	1			**1**
AB				
AB’				

4) Simplify the expression – using K-Map

E = A’B’CD’ + A’BC’D’ + A’BCD’ + AB’C’D’ + AB’CD’ + ABC’D’ + ABCD’

	C’D’	C’D	CD	CD’
A’B’				1
A’B	1			1
AB				
AB’	**1**			

4) Simplify the expression – using K-Map

E = A’B’CD’ + A’BC’D’ + A’BCD’ + AB’C’D’ + AB’CD’ + ABC’D’ + ABCD’

	C’D’	C’D	CD	CD’
A’B’				1
A’B	1			1
AB				
AB’	1			**1**

4) Simplify the expression – using K-Map

E = A’B’CD’ + A’BC’D’ + A’BCD’ + AB’C’D’ + AB’CD’ + ABC’D’ + ABCD’

	C’D’	C’D	CD	CD’
A’B’				1
A’B	1			1
AB	1			
AB’	1			1

4) Simplify the expression – using K-Map

E = A’B’CD’ + A’BC’D’ + A’BCD’ + AB’C’D’ + AB’CD’ + ABC’D’ + ABCD’

	C’D’	C’D	CD	CD’
A’B’				1
A’B	1			1
AB	1			**1**
AB’	1			1

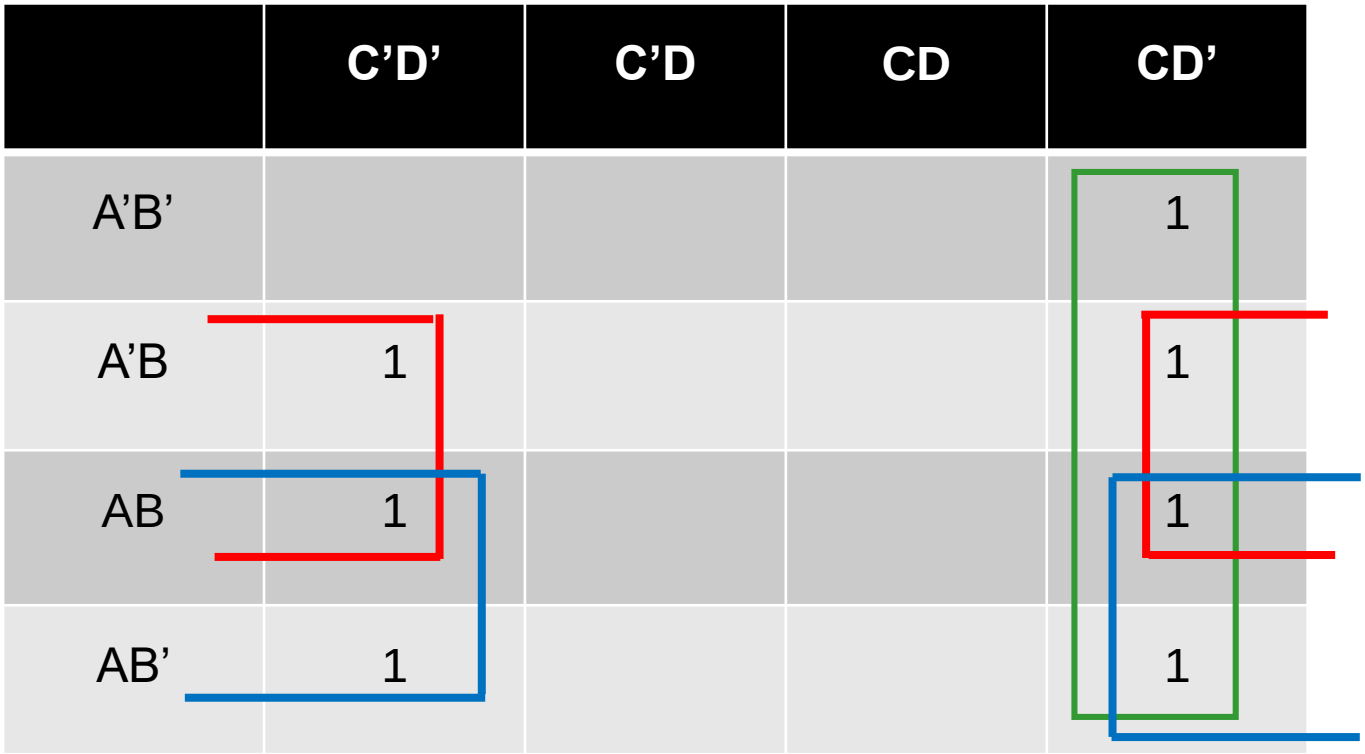
4) Simplify the expression – using K-Map

E = A’B’CD’ + A’BC’D’ + A’BCD’ + AB’C’D’ + AB’CD’ + ABC’D’ + ABCD’

	C’D’	C’D	CD	CD’
A’B’				1
A’B	1			1
AB	1			1
AB’	1			1

4) Simplify the expression – using K-Map

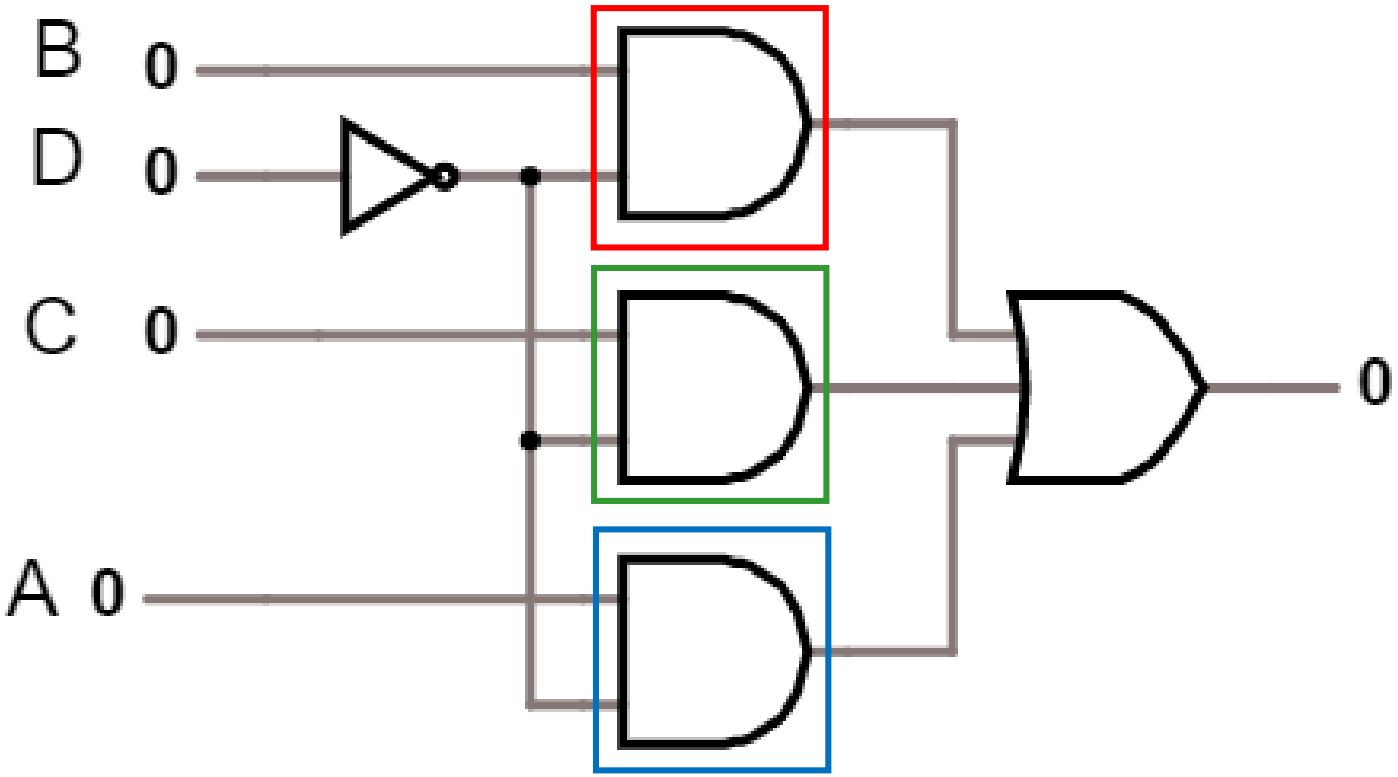
E = A’B’CD’ + A’BC’D’ + A’BCD’ + AB’C’D’ + AB’CD’ + ABC’D’ + ABCD’



Simplified expression is BD’ + CD’ + AD’

5) Implement circuit:

Simplified expression is BD’ + CD’ + AD’



Week 7 Lecture 12a

- Combinational circuit
- Assignment 2 – submit before midnight Monday
- Course web page:

  https://ecs.wgtn.ac.nz/Courses/XMUT101_2021T1/

- kerese@ecs.vuw.ac.nz