CS2100
COMPUTER ORGANISATION

http://www.comp.nus.edu.sg/~cs2100/

CS2100
Recitation 8

Combinational Circuits

17 March 2025

Aaron Tan

NUS National University of Singapore | School of Computing

Contents

Feedback

- Students expressed wish for shorter recitation sessions
- I’ve cut down the materials, replacing with problem solving questions using past years’ exam questions.

ANDROID
NANDROID
NOTDROID
ORDROID

SUMMARY

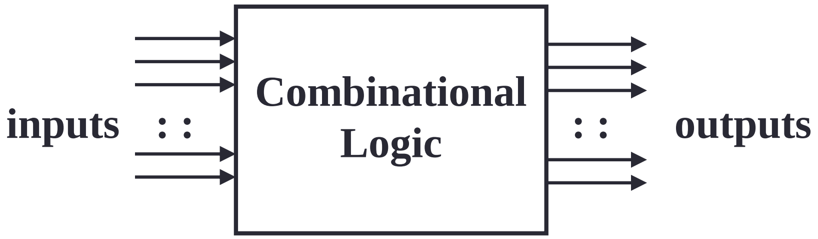
Design Methods

1. Introduction

- Two classes of logic circuits

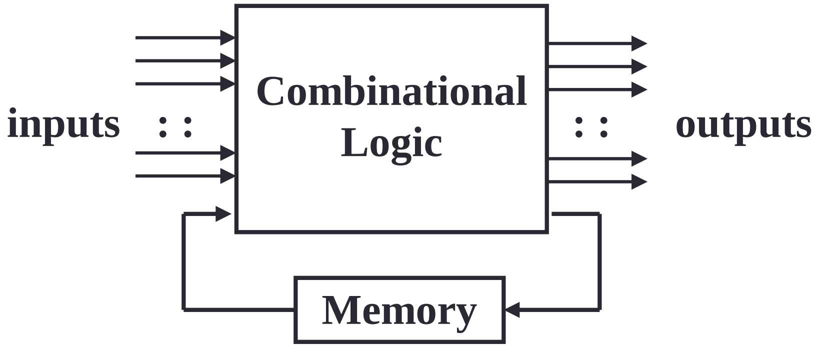
- Combinational Circuit
  - Each output depends entirely on the immediate (present) inputs.

This week and next week.



- Sequential Circuit
  - Each output depends on both present inputs and state.

Week 11.



- Design methods
  - Gate-level design method (with logic gates)
  - Block-level design method (with functional blocks)

4. BCD to Excess-3 Code Converter (1/3)

Other decimal codes we used to cover:

Digit	BCD code (8421 code)	Excess-3 code	84-2-1 code	2421 code
0	0000	0011	0000	0000
1	0001	0100	0111	0001
2	0010	0101	0110	0010
3	0011	0110	0101	0011
4	0100	0111	0100	0100
5	0101	1000	1011	1011
6	0110	1001	1010	1100
7	0111	1010	1001	1101
8	1000	1011	1000	1110
9	1001	1100	1111	1111

Note: Decimal codes are only valid for the ten decimal digits 0 through 9.

4. BCD to Excess-3 Code Converter (2/3)

- Truth table:

	BCD				*Excess-3*			
	A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0
10	1	0	1	0	X	X	X	X
11	1	0	1	1	X	X	X	X
12	1	1	0	0	X	X	X	X
13	1	1	0	1	X	X	X	X
14	1	1	1	0	X	X	X	X
15	1	1	1	1	X	X	X	X

- K-maps:

W

AB \ CD	00	01	11	10
00	0	0	0	0
01	0	1	1	1
11	X	X	X	X
10	1	1	X	X

X

AB \ CD	00	01	11	10
00	0	1	1	1
01	1	0	0	0
11	X	X	X	X
10	0	1	X	X

Y

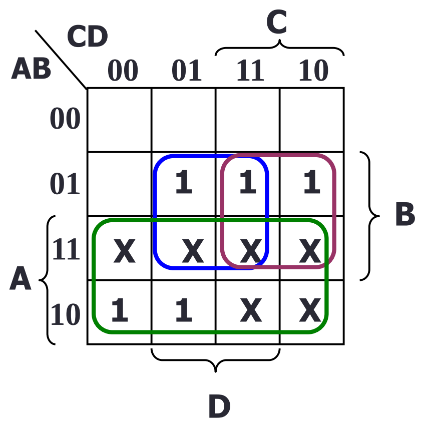
AB \ CD	00	01	11	10
00	1	0	1	0
01	1	0	1	0
11	X	X	X	X
10	1	0	X	X

Z

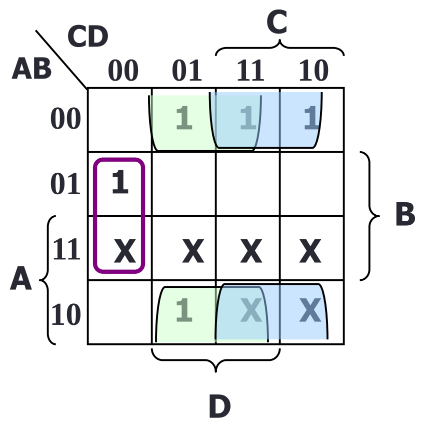
AB \ CD	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	X	X	X	X
10	1	0	X	X

4. BCD to Excess-3 Code Converter (3/3)

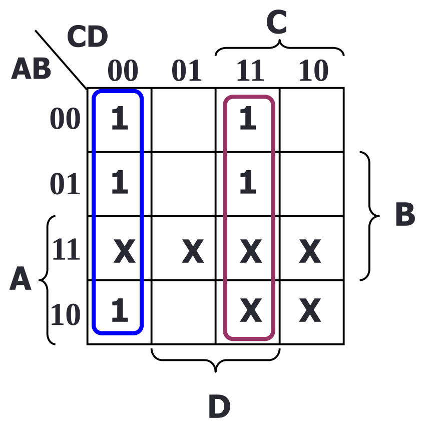
W



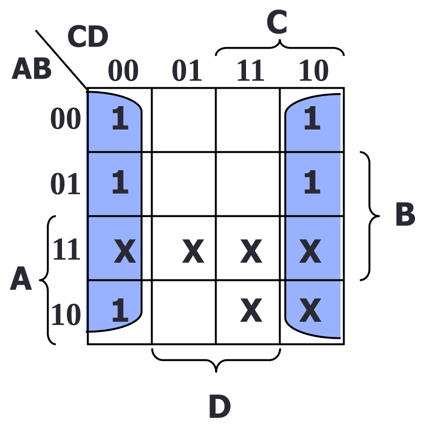
X



Y



Z



W = A + B·C + B·D

X = B'·C + B'·D + B·C'·D'

Y = C.D + C'.D'

Z = D'

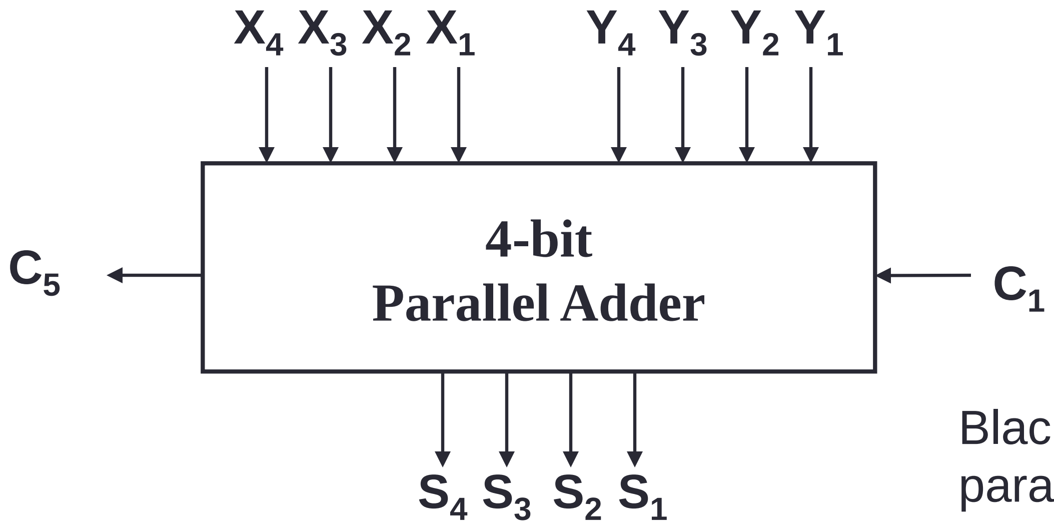
Note: For illustration purpose here, we left out the 0s on the K-map for brevity. In practice, you must fill in all cells.

5. Block-Level Design

- More complex circuits can also be built using block-level method.
- In general, block-level design method (as opposed to gate-level design) relies on algorithms or formulae of the circuit, which are obtained by decomposing the main problem to sub-problems recursively (until small enough to be directly solved by blocks of circuits).
- First example shows how to create a 4-bit parallel adder using block-level design.
- Using 4-bit parallel adders as building blocks, we can create the following:
  1. BCD-to-Excess-3 Code Converter
  2. 16-bit Parallel Adder

5. 4-bit Parallel Adder (1/4)

- Consider a circuit to add two 4-bit numbers together and a carry-in, to produce a 5-bit result.



Black-box view of 4-bit parallel adder

- 5-bit result is sufficient because the largest result is:

$$1111_2 + 1111_2 + 1_2 = 11111_2$$

5. 4-bit Parallel Adder (2/4)

- SSI design (gate-level design) technique should not be used here.
- Truth table for 9 inputs is too big: $2^9 = 512$ rows!

$X_4X_3X_2X_1$	$Y_4Y_3Y_2Y_1$	C_1	C_5	$S_4S_3S_2S_1$
0 0 0 0	0 0 0 0	0	0	0 0 0 0
0 0 0 0	0 0 0 0	1	0	0 0 0 1
0 0 0 0	0 0 0 1	0	0	0 0 0 1
…	…	…	…	…
0 1 0 1	1 1 0 1	1	1	0 0 1 1
…	…	…	…	…
1 1 1 1	1 1 1 1	1	1	1 1 1 1

- Simplification becomes too complicated!

5. 4-bit Parallel Adder (3/4)

- Alternative design possible.
- Addition formula for each pair of bits (with carry in),

  $C_{i+1}S_i = X_i + Y_i + C_i$

  has the same function as a full adder:

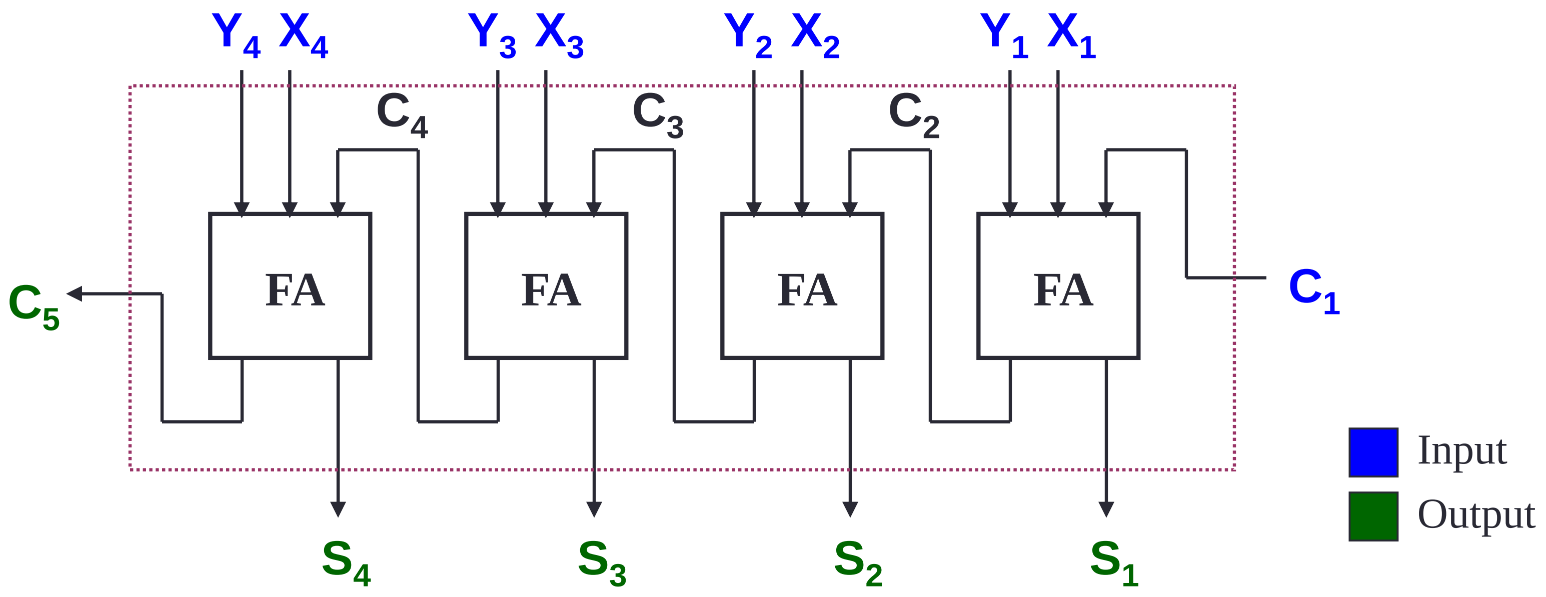
  $C_{i+1} = X_i \cdot Y_i + (X_i \oplus Y_i) \cdot C_i$

  $S_i = X_i \oplus Y_i \oplus C_i$

C =		1	1	0	0
X =		1	0	1	0
Y =		1	1	1	1
X + Y =	1	1	0	0	1

5. 4-bit Parallel Adder (4/4)

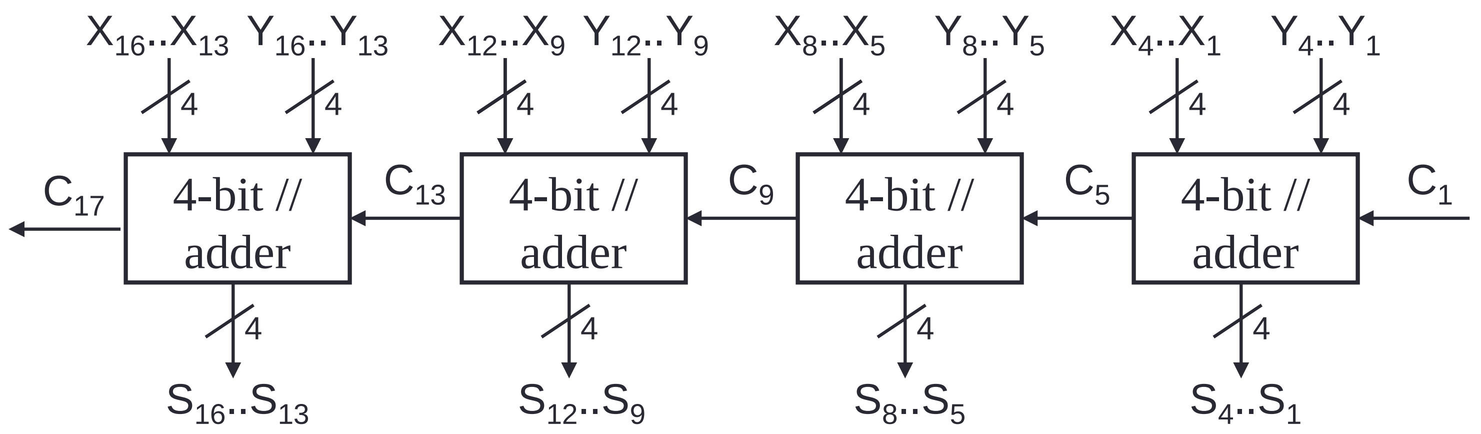
- Cascading 4 full adders via their carries, we get:



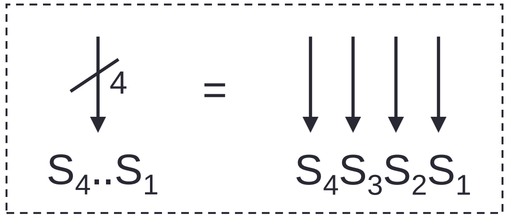
- Note that carry is propagated by cascading the carry from one full adder to the next.
- Called Parallel Adder because inputs are presented simultaneously (in parallel). Also called Ripple-Carry Adder.

5. 16-bit Parallel Adder

- Larger parallel adders can be built from smaller ones.
- Example: A 16-bit parallel adder can be constructed from four 4-bit parallel adders:



A 16-bit parallel adder



5. BCD to Excess-3 Converter: Revisit (1/2)

- Excess-3 code can be converted from BCD code using truth table:
- Gate-level design can be used since only 4 inputs.
- However, alternative design is possible.
- Use problem-specific formula:

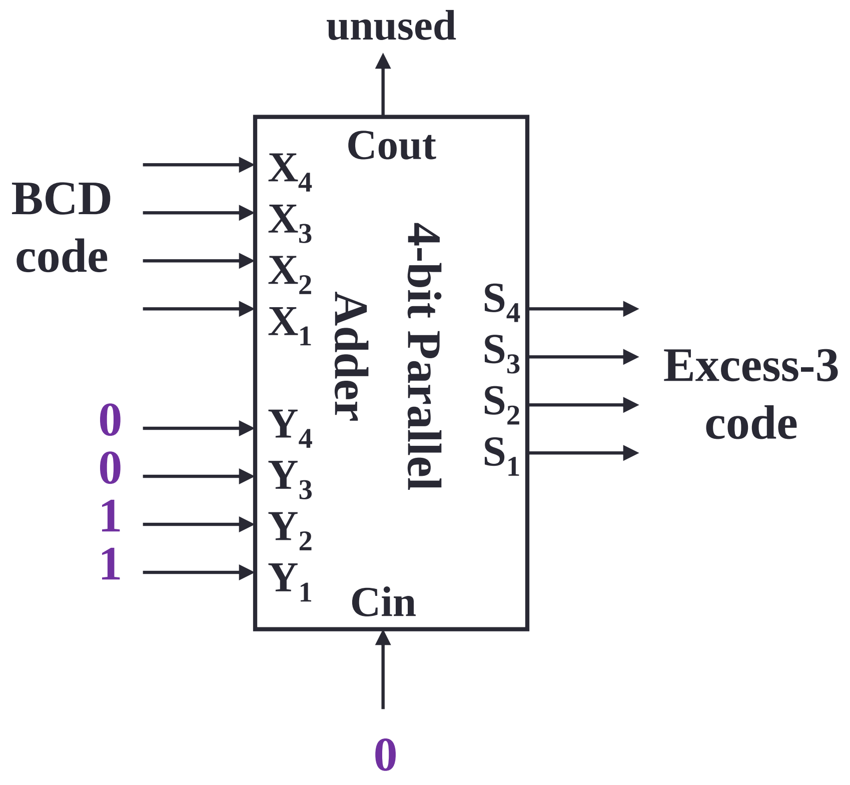
  Excess-3 code
  = BCD Code + 0011_2

	BCD				*Excess-3*			
	A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0
10	1	0	1	0	X	X	X	X
11	1	0	1	1	X	X	X	X
12	1	1	0	0	X	X	X	X
13	1	1	0	1	X	X	X	X
14	1	1	1	0	X	X	X	X
15	1	1	1	1	X	X	X	X

5. BCD to Excess-3 Converter: Revisit (2/2)

- Block-level circuit:

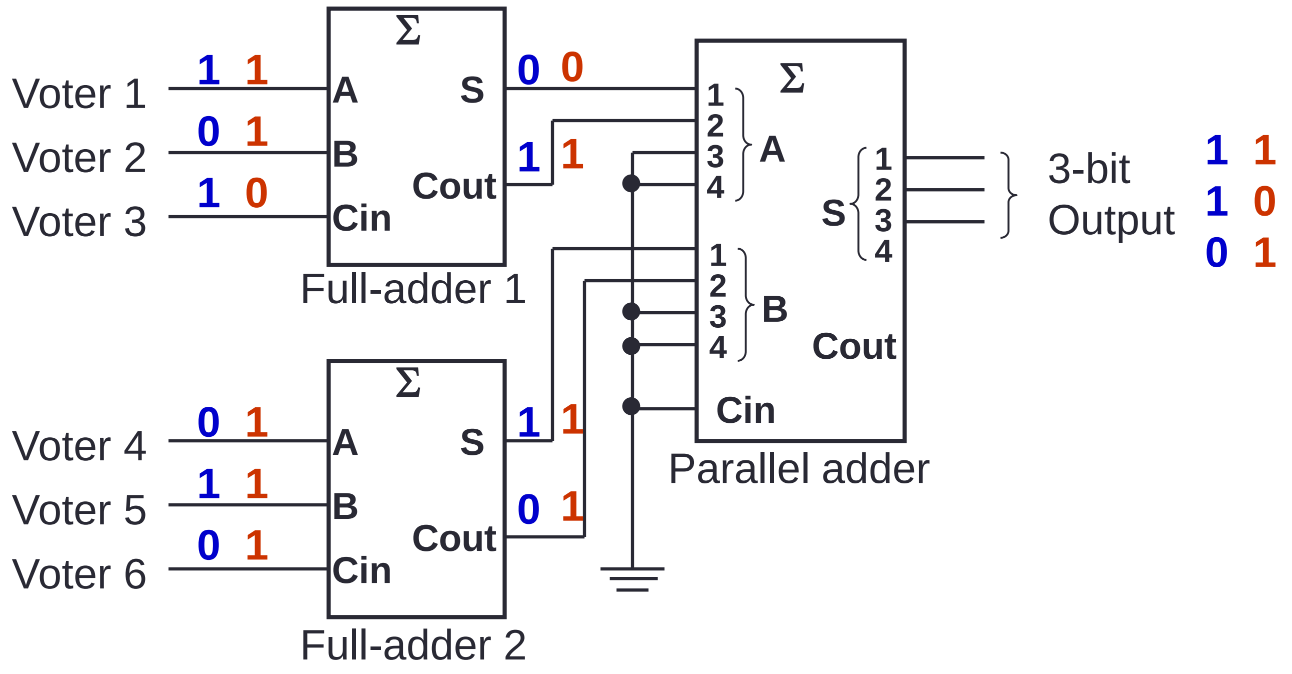
A BCD to Excess-3 Code Converter



Note: In the lab, input 0 (low) is connected to GND, 1 (high) to Vcc.

7. Example: 6-Person Voting System

- Application: 6-person voting system.
  - Use FAs and a 4-bit parallel adder.
  - Each FA can sum up to 3 votes.



SUMMARY

Magnitude Comparator

8. Magnitude Comparator (1/4)

- Magnitude comparator: compares 2 unsigned values A and B, to check if A>B, A=B, or A<B.
- To design an *n*-bit magnitude comparator using classical method, it would require 2^{2n} rows in truth table!
- We shall exploit regularity in our design.
- Question: How do we compare two 4-bit unsigned values A ($a_3a_2a_1a_0$) and B ($b_3b_2b_1b_0$)?

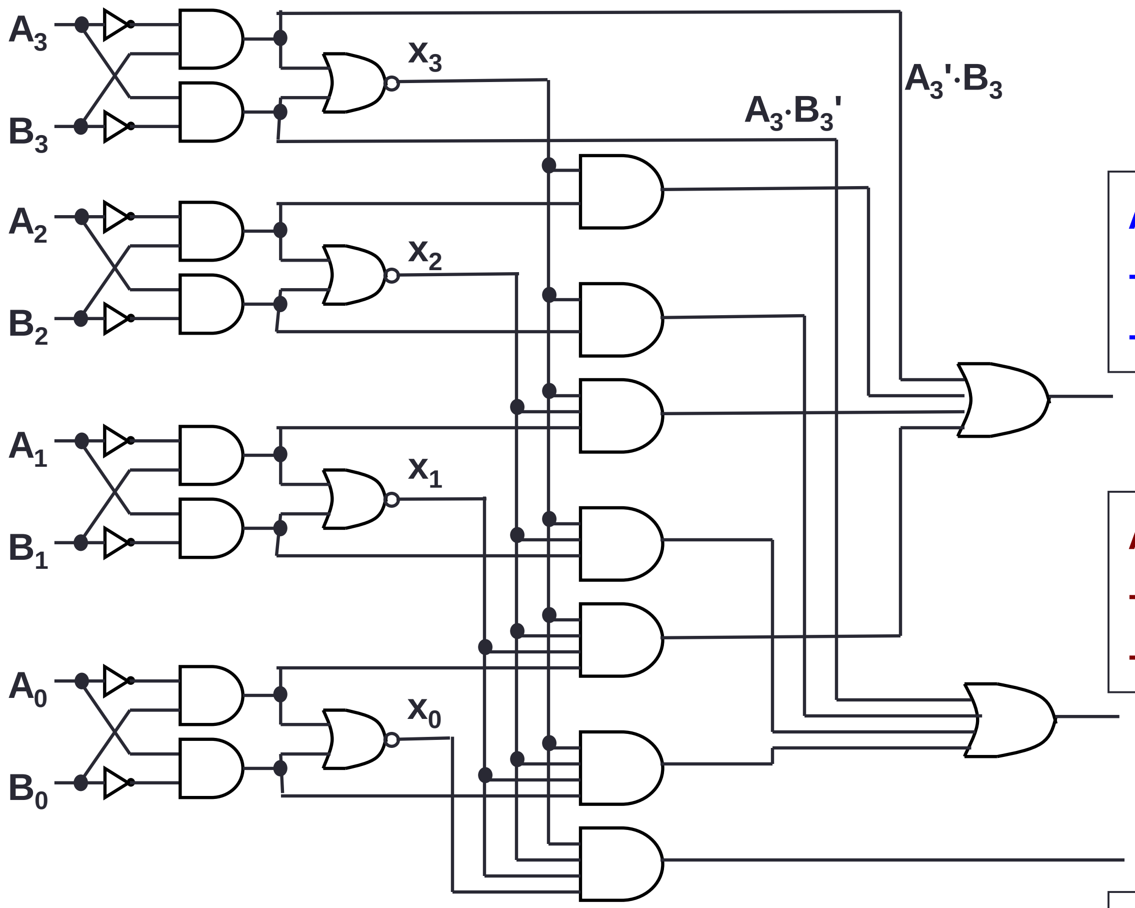
  If ($a_3 > b_3$) then A > B

  If ($a_3 < b_3$) then A < B

  If ($a_3 = b_3$) then if ($a_2 > b_2$) …

8. Magnitude Comparator (2/4)

Let $A = A_3A_2A_1A_0$, $B = B_3B_2B_1B_0$; $x_i = A_i \cdot B_i + A_i' \cdot B_i'$



$A_3' \cdot B_3 + x_3 \cdot A_2' \cdot B_2$
$+ x_3 \cdot x_2 . A_1' \cdot B_1$
$+ x_3 \cdot x_2 \cdot x_1 \cdot A_0' \cdot B_0$

(A < B)

$A_3 \cdot B_3' + x_3 \cdot A_2 \cdot B_2'$
$+ x_3 \cdot x_2 \cdot A_1 \cdot B_1'$
$+ x_3 \cdot x_2 \cdot x_1 \cdot A_0 \cdot B_0'$

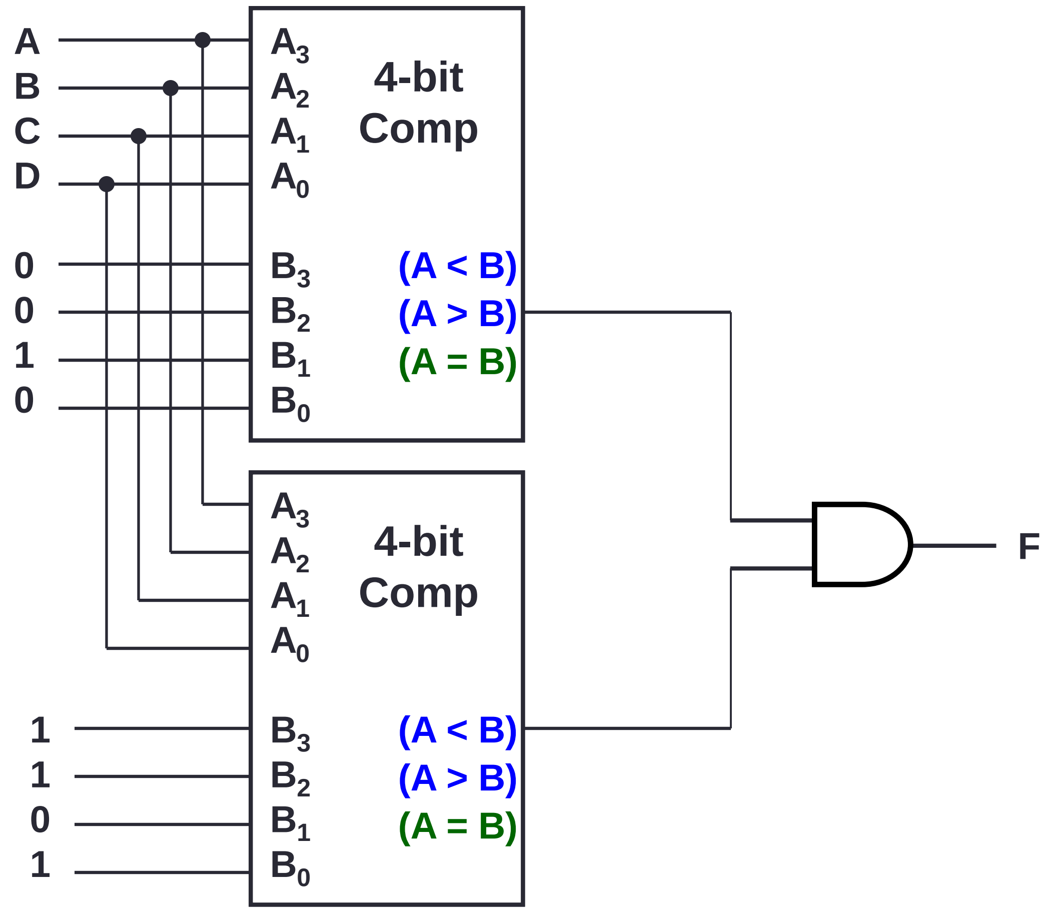
(A > B)

(A = B)

$x_3 \cdot x_2 \cdot x_1 \cdot x_0$

8. Magnitude Comparator (4/4)

- A function F accepts a 4-bit binary value ABCD, and returns 1 if $3 \leq ABCD \leq 12$, or 0 otherwise. How would you implement F using magnitude comparators and a suitable logic gate?



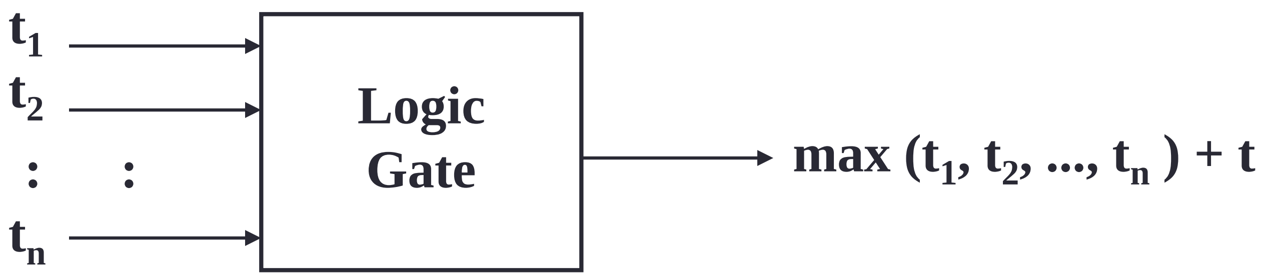
SUMMARY

Circuit Delays

9. Circuit Delays (1/5)

- Given a logic gate with delay t. If inputs are stable at times $t_1, t_2, \ldots, t_n$, then the earliest time in which the output will be stable is:

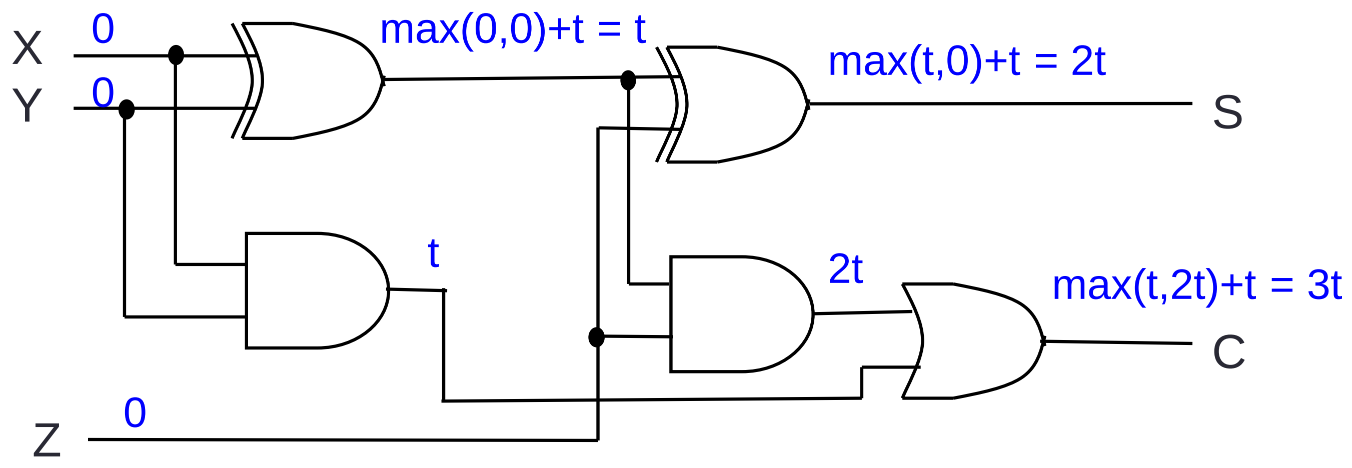
$$\max(t_1, t_2, \ldots, t_n) + t$$



- To calculate the delays of all outputs of a combinational circuit, repeat above rule for all gates.

9. Circuit Delays (2/5)

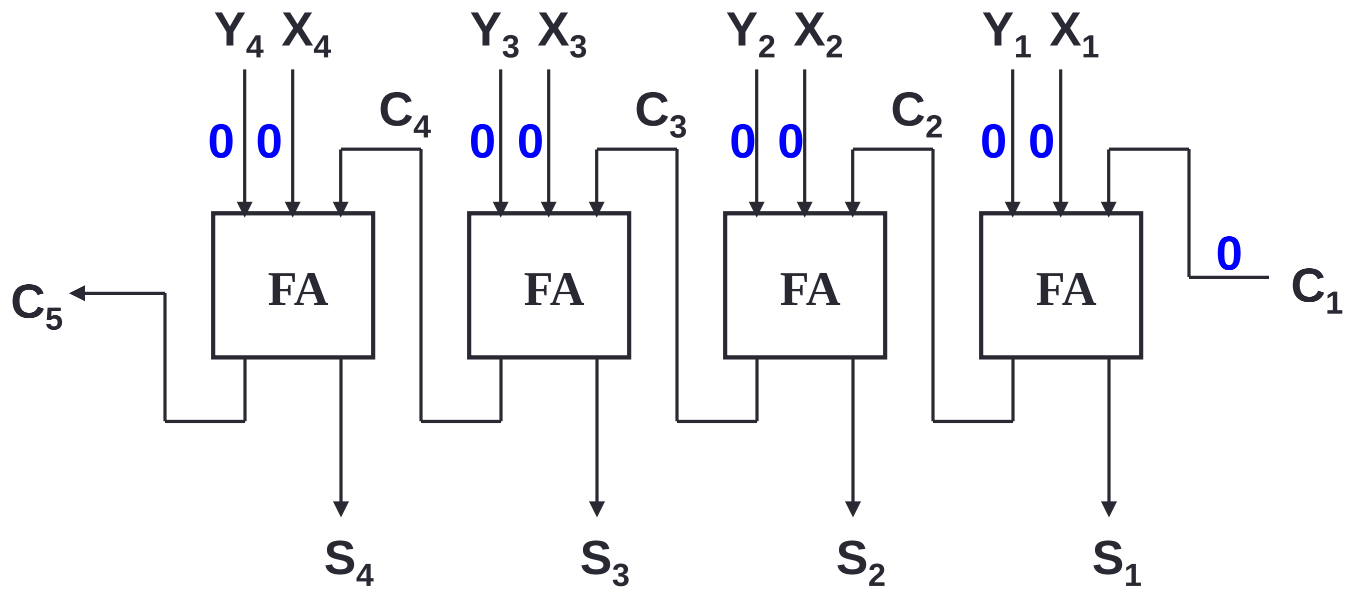
- As a simple example, consider the full adder circuit where all inputs are available at time 0. Assume each gate has delay t.



- Outputs **S** and **C** experience delays of **2t** and **3t** respectively.

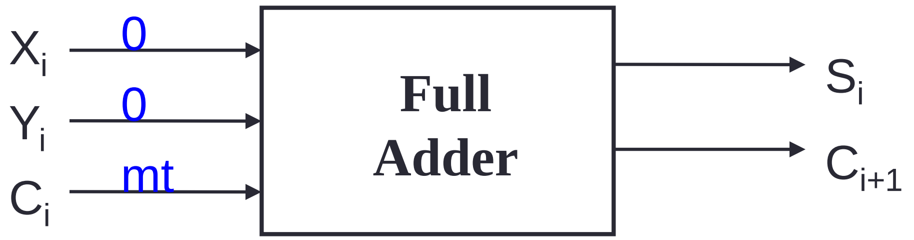
9. Circuit Delays (3/5)

- More complex example: 4-bit parallel adder.



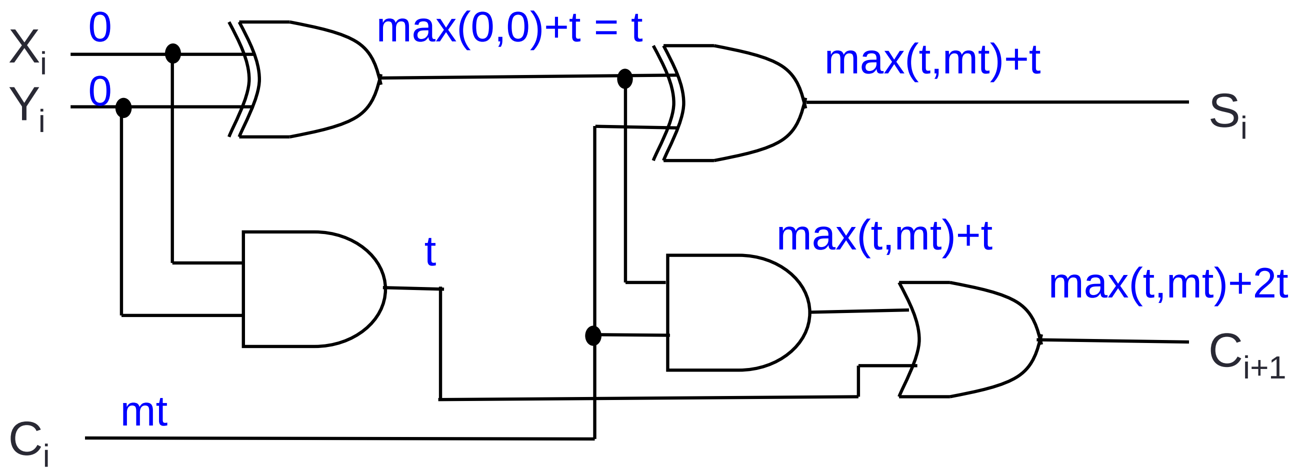
9. Circuit Delays (4/5)

- Analyse the delay for the repeated block.

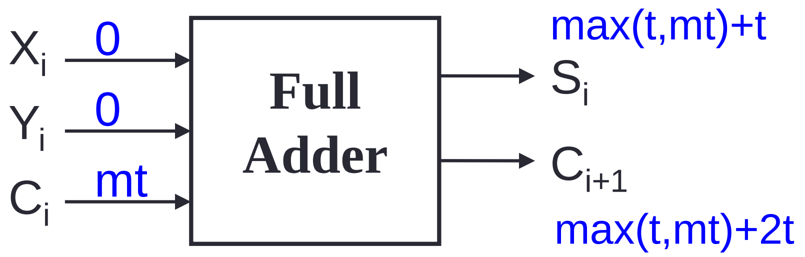


where X_i, Y_i are stable at 0t, while C_i is assumed to be stable at mt.

- Performing the delay calculation:



9. Circuit Delays (5/5)



- Calculating:

  When i=1, m=0; $S_1 = 2t$ and $C_2 = 3t$

  When i=2, m=3; $S_2 = 4t$ and $C_3 = 5t$

  When i=3, m=5; $S_3 = 6t$ and $C_4 = 7t$

  When i=4, m=7; $S_4 = 8t$ and $C_5 = 9t$

- In general, an *n*-bit ripple-carry parallel adder will experience the following delay times:

  $S_n = ((n - 1)2 + 2)\ t$

  $C_{n+1} = ((n - 1)2 + 3)\ t$

- Propagation delay of ripple-carry parallel adders is proportional to the number of bits it handles.

- Maximum delay: $((n - 1)2 + 3)\ t$

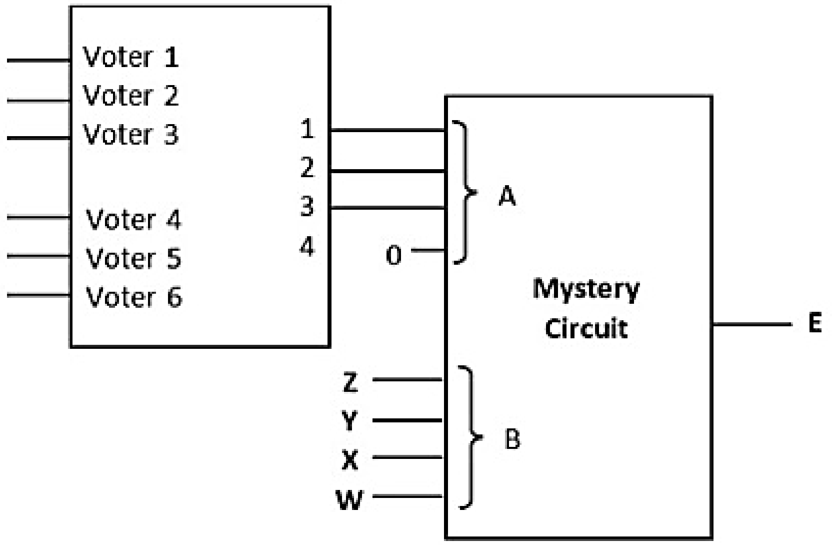
QUIZZES

19. Combinational Circuits Quiz 1
20. Combinational Circuits Quiz 2

19. Combinational Circuits Quiz 1 Q1

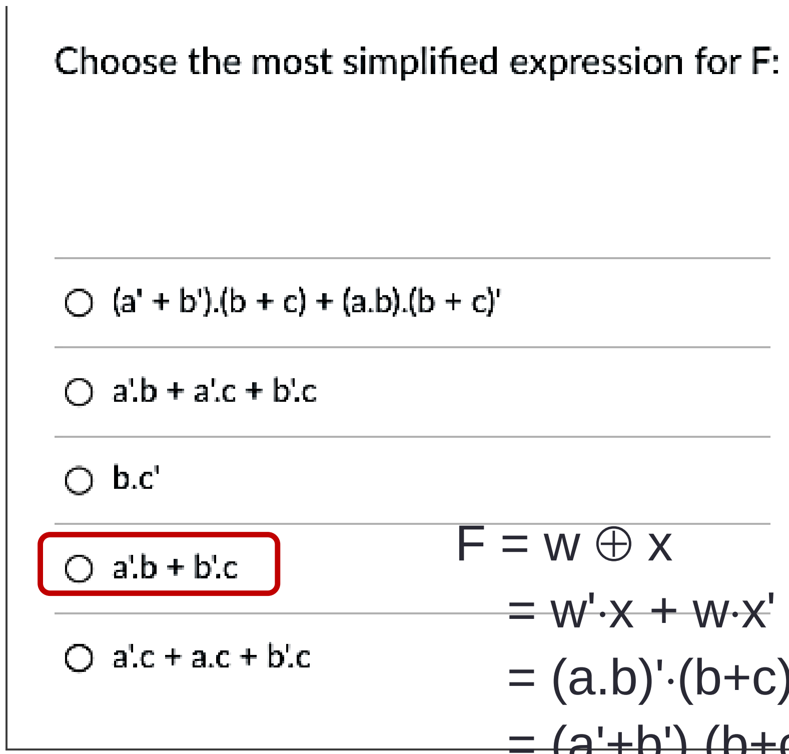
Question 1 **1 pts**

We return to the 6 Person Voting System in the lecture. Rather than producing the sum of the votes, what we want now is to produce a circuit that outputs a 1 to E when at least four persons have voted "1" (i.e. "Yes"). The 6-person voting circuit is represented by a single block below. Please see Lecture 17 Slide 29/30 for the actual circuit:



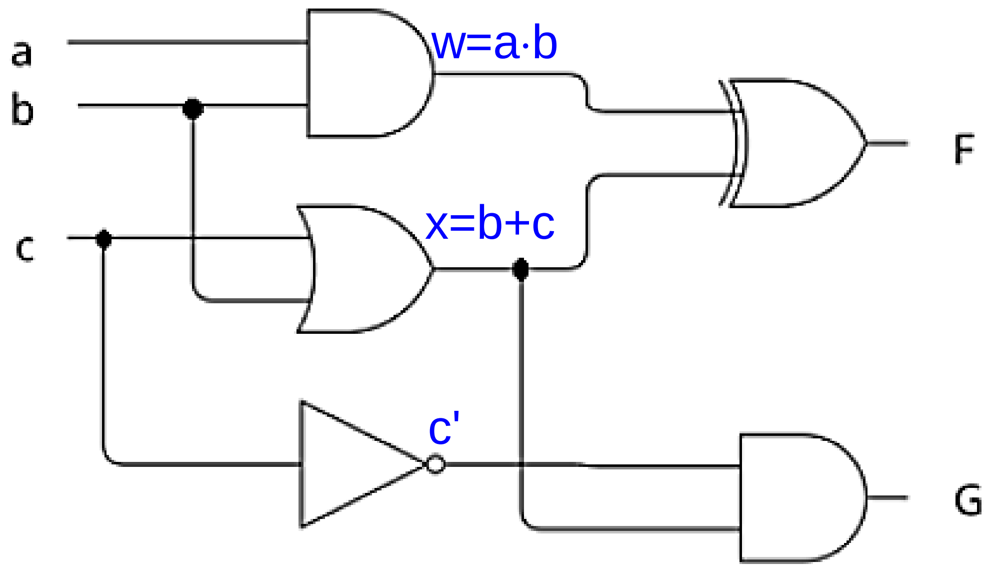
- ○ The mystery circuit is a 4-bit adder, WXYZ = 0b0011 and E comes from the most significant bit of the sum.
- [✓ ○ The mystery circuit is a 4-bit comparator, WXYZ =0b0011, and E comes from the (A > B) output of the comparator.]
- ○ The mystery circuit is a 4-bit adder, WXYZ = 0b1100, and E comes from the most significant bit of the sum.
- ○ The mystery circuit is a 4-bit comparator, WXYZ=0b0100, and E comes from the (A = B) output of the comparator.
- ○ The mystery circuit is a 4-bit adder, WXYZ=0b0011, and E comes from the least significant bit of the sum.

20. Combinational Circuits Quiz 2 Q1



Choose the most simplified expression for F:

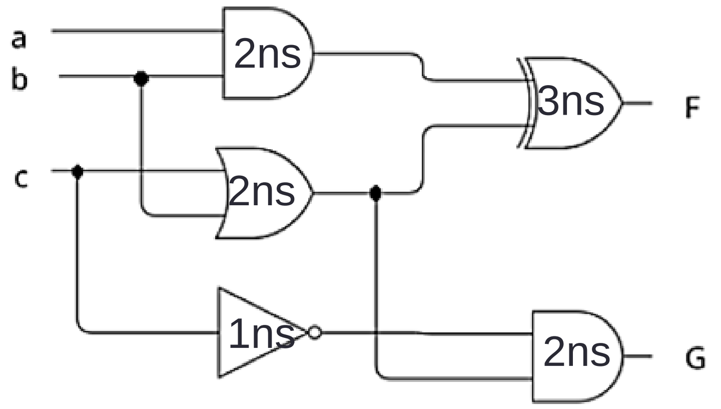
- ○ (a' + b').(b + c) + (a.b).(b + c)'
- ○ a'.b + a'.c + b'.c
- ○ b.c'
- [✓ ○ a'.b + b'.c]
- ○ a'.c + a.c + b'.c



F = w ⊕ x
= w'·x + w·x'
= (a.b)'·(b+c) + (a.b)·(b+c)'
= ~~(a'+b').(b+c)~~ + (a.b).(b'.c')
= a'.b + a'.c + b'.b + b'.c + a.b.b'.c'
= a'.b + a'.c + 0 + b'.c + 0
= a'.b + a'.c + b'.c
= a'.b + b'.c

Concensus Theorem:
X·Y + X'·Z + Y·Z = X·Y + X'·Z

20. Combinational Circuits Quiz 2 Q2, Q3



Question 2

Given that AND and OR gates have a propagation delay of 2 ns, NOT gates have a propagation delay of 1 ns and XOR gates have a propagation delay of 3 ns,

What is the propagation delay in ns at output F? 5ns

Question 3

Given the same propagation delays as before, what is the propagation delay in ns at output G? 4ns

PAST YEARS' QUESTIONS

Combinational Circuits

Past years' exam questions

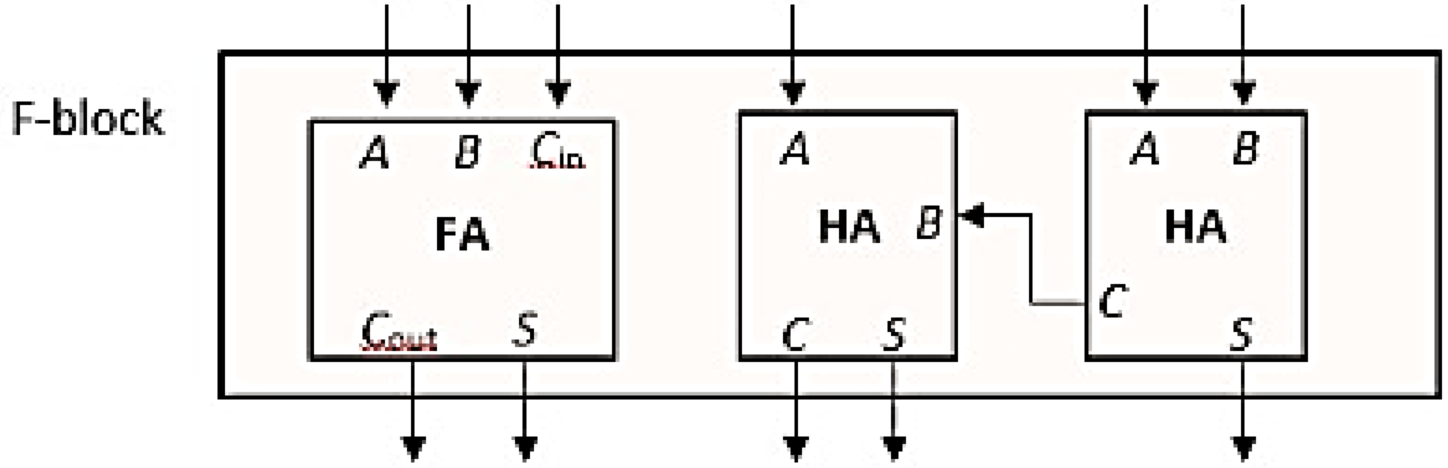
- AY2020/21 Sem2 Q13(d)
- AY2021/22 Sem1 Q13(a)
- AY2021/22 Sem2 Q14(b)(c)
- AY2023/24 Sem1 Q17(a)(b)(d)

Assumption

- Logical constants 0 and 1 are available, but not complemented literals.

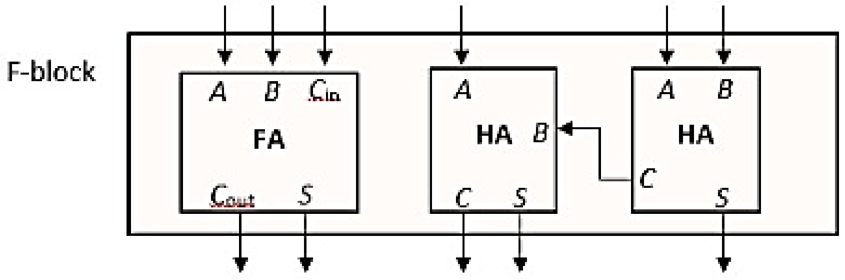
AY2020/21 Sem2 Q13(d)

(d) The F-block contains two half adders and a full adder as shown below. Using a single F-block, without any additional logic gates, design a circuit that takes in a 3-bit unsigned binary number $X = X_2X_1X_0$ and a one-bit value y to generate the output $Z_3Z_2Z_1Z_0$ which is a binary number representing the value $X+5y$. You may only connect inputs to the 6 inwards arrows and use the 5 outwards arrows for your outputs. [4 marks]



AY2020/21 Sem2 Q13(d)

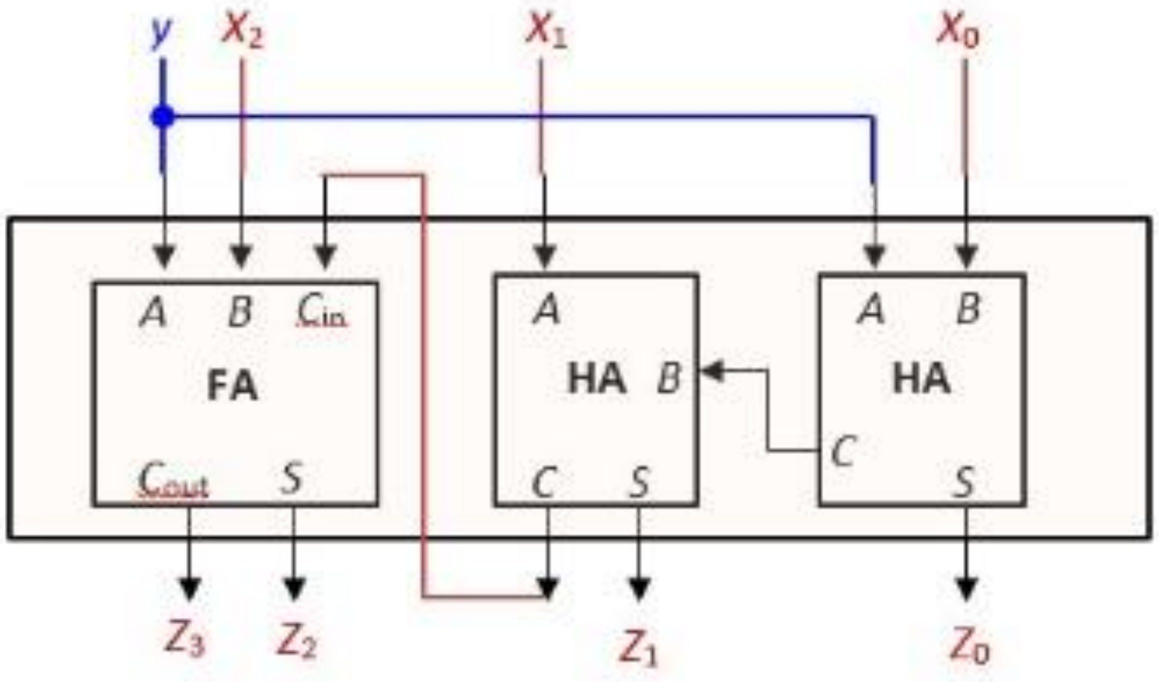
(d) The F-block contains two half adders and a full adder as shown below. Using a single F-block, without any additional logic gates, design a circuit that takes in a 3-bit unsigned binary number $X=X_2X_1X_0$ and a one-bit value y to generate the output $Z_3Z_2Z_1Z_0$ which is a binary number representing the value $X+5y$. You may only connect inputs to the 6 inwards arrows and use the 5 outwards arrows for your outputs. [4 marks]



$Z = X + 5y$

When y=0: $Z_3Z_2Z_1Z_0 = X_2X_1X_0 + 000$

When y=1: $Z_3Z_2Z_1Z_0 = X_2X_1X_0 + 101$

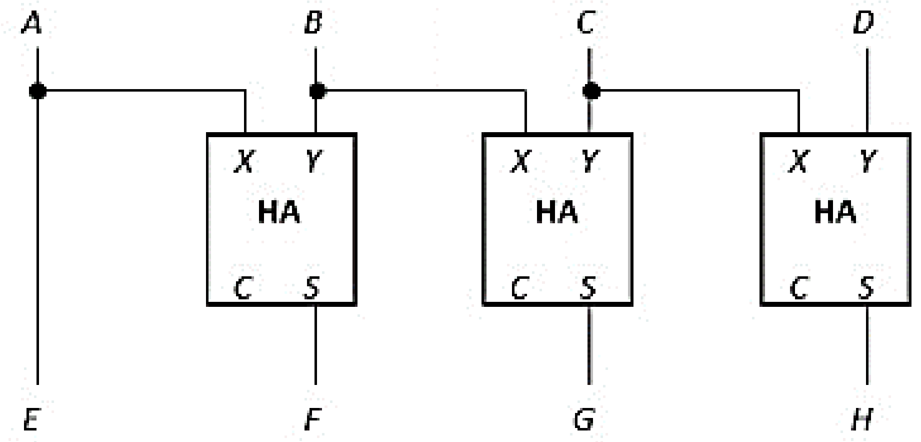


AY2021/22 Sem1 Q13(a)

(a) The circuit below shows 3 half-adders. Each half adder takes inputs *X* and *Y* and produces outputs *C* (carry) and *S* (sum). The circuit takes in a 4-bit input *ABCD* and generates a 4-bit output *EFGH*.

Write out the Σm notation for the functions *E*(*A*,*B*,*C*,*D*), *F*(*A*,*B*,*C*,*D*), *G*(*A*,*B*,*C*,*D*), and *H*(*A*,*B*,*C*,*D*).

[4 marks]



E(A,B,C,D) = Σm(8,9,10,11,12,13,14,15)

F(A,B,C,D) = Σm(4,5,6,7,8,9,10,11)

G(A,B,C,D) = Σm(2,3,4,5,10,11,12,13)

H(A,B,C,D) = Σm(1,2,5,6,9,10,13,14)

A	B	C	D	E	F	G	H
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

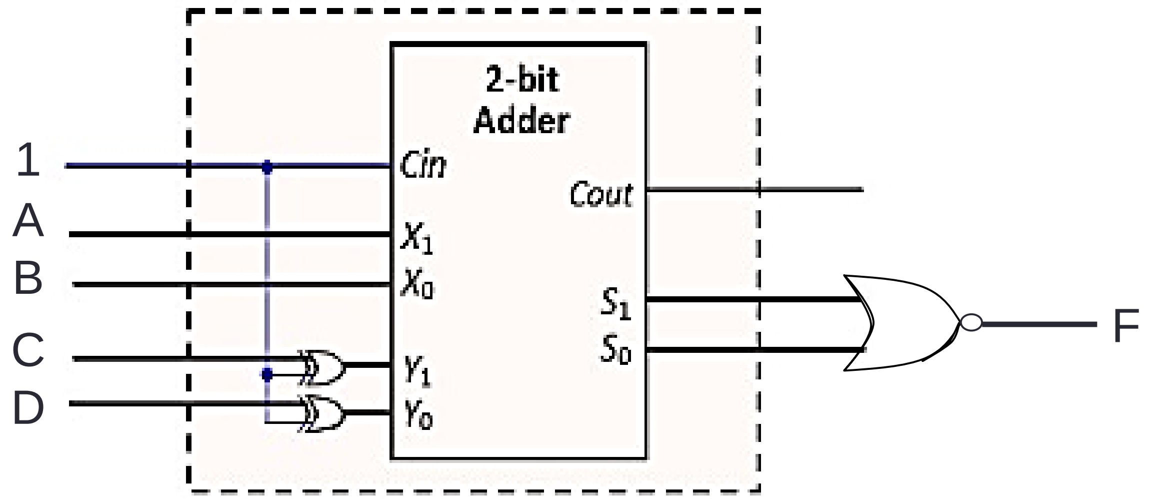
AY2021/22 Sem2 Q14(b)

Part (a) answer: $F = \sum m(0, 5, 10, 15)$

A	B	C	D	F
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

(b) The circuit below comprises a 2-bit parallel adder and 2 XOR gates. The circuit is housed inside a chip so the only connections available are those that extend out of the dotted box.

Using this circuit and one additional logic gate (inverter, AND, OR, NAND, NOR, XOR, or XNOR), implement the function F in part (a) above. [4 marks]

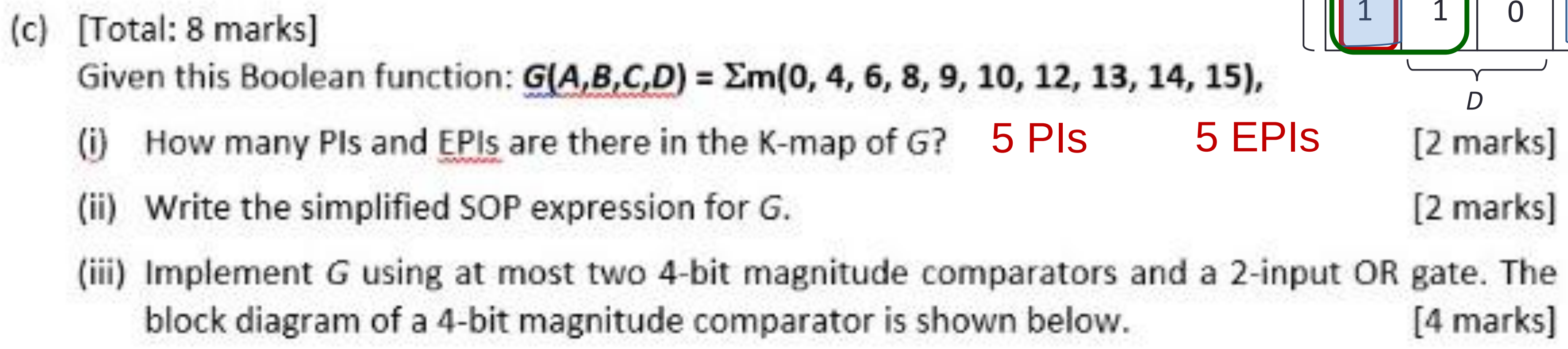


What is this circuit?

It is a 2-bit adder-cum-substractor.
If Cin is set to 0, it performs $X + Y$.
If Cin is 1, it performs $X – Y$.

Since F=1 when AB=CD, or AB – CD = 0, the solution is…

AY2021/22 Sem2 Q14(c)

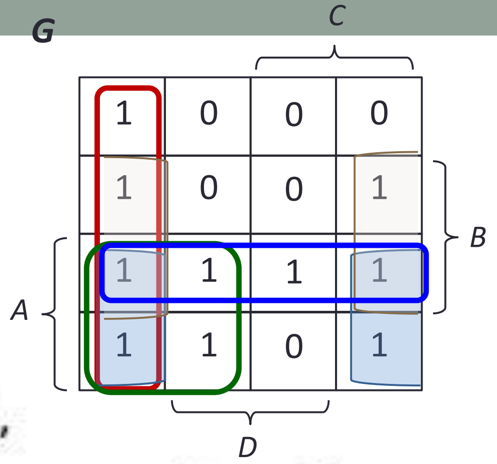


(c) [Total: 8 marks]
Given this Boolean function: **G(A,B,C,D) = Σm(0, 4, 6, 8, 9, 10, 12, 13, 14, 15),**

(i) How many PIs and EPIs are there in the K-map of *G*? 5 PIs 5 EPIs [2 marks]

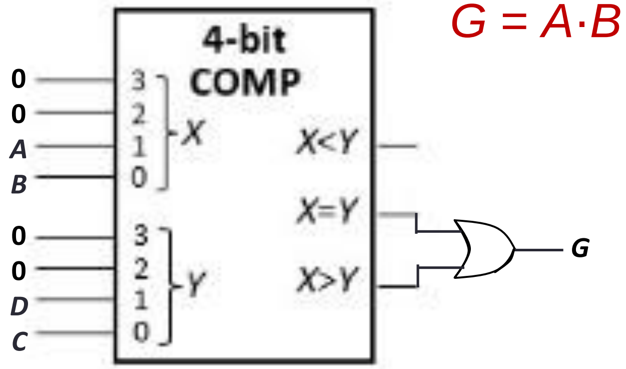
(ii) Write the simplified SOP expression for *G*. [2 marks]

(iii) Implement *G* using at most two 4-bit magnitude comparators and a 2-input OR gate. The block diagram of a 4-bit magnitude comparator is shown below. [4 marks]



A	B	C	D	G
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

$G = A{\cdot}B + C'{\cdot}D' + A{\cdot}C' + B{\cdot}D' + A{\cdot}D'$



Observation: $G = (AB \geq DC)$

(Other answers possible. Eg: $ABC \geq DCB$, $ABCD \geq DCBA$).

AY2023/24 Sem1 Q17(a)

Q17. Combinational circuits [13 marks]

(a) Given the following Boolean function $P(W,X,Y,Z)$, where d's are don't cares:

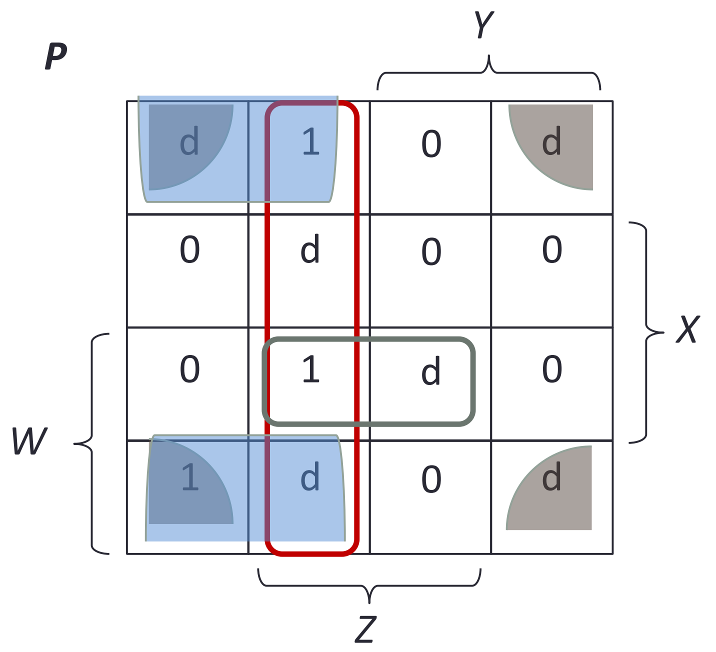
$$P(W,X,Y,Z) = \Sigma m(1,8,13) + d(0,2,5,9,10,15).$$

(i) How many prime implicants are there in the K-map of P?

(ii) How many essential prime implicants are there in the K-map of P?

4 PIs: $Y'{\cdot}Z$, $X'{\cdot}Y'$, $X'{\cdot}Z'$, $W{\cdot}X{\cdot}Z$

0 EPI



AY2023/24 Sem1 Q17(b)

(b) Given the following Boolean function *Q*(*W*,*X*,*Y*,*Z*), where *d*'s are don't cares:

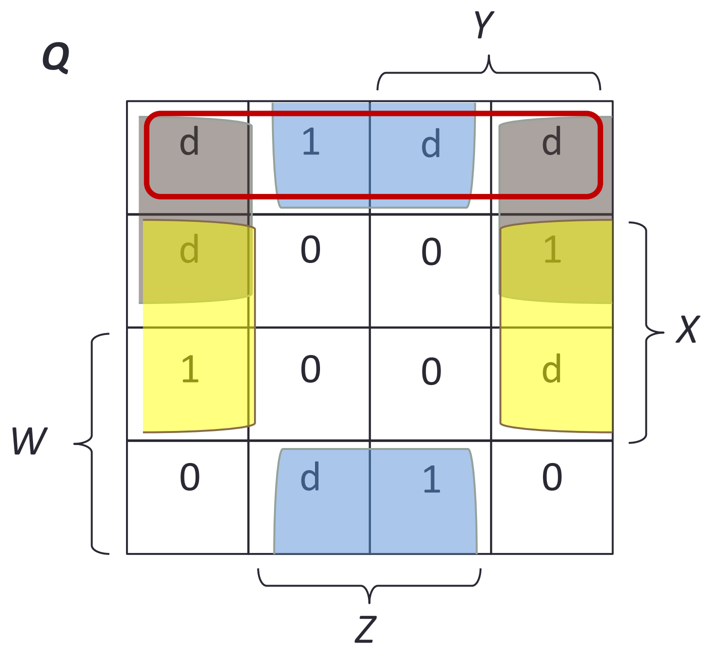
$$Q(W,X,Y,Z) = \Pi M(5,7,8,10,13,15) \cdot d(0,2,3,4,9,14).$$

(i) How many prime implicants are there in the K-map of *Q*?

(ii) How many essential prime implicants are there in the K-map of *Q*?

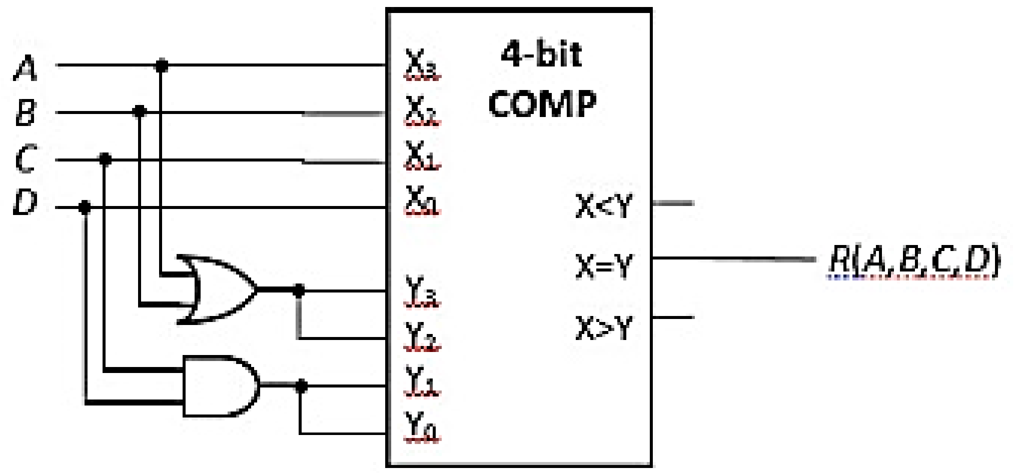
4 PIs: W'·X', W'·Z', X'·Z, X·Z'

2 EPIs: X'·Z, X·Z'



AY2023/24 Sem1 Q17(d)

(d) A Boolean function *R(A,B,C,D)* is implemented with a 4-bit magnitude comparator, an OR gate and an AND gate as shown below:



What is *R(A,B,C,D)* in Σm notation?

$R(A,B,C,D) = \Sigma m(\mathbf{0,3,12,15})$.

A	B	C	D	R
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

NEXT WEEK

MSI Components:

- Decoders, Encoders, Multiplexers, Demultiplexers

Past years' exam questions:

- AY2016/17 Sem2 Q2
- AY2017/18 Sem2 Q3(a)(b)(c)
- AY2018/19 Sem2 Q5(c)
- AY2019/20 Sem2 Q5(a)(b)

End of File