CS2100 (AY2020/21 Semester 2)
Assignment #2

PLEASE READ THE FOLLOWING INSTRUCTIONS VERY CAREFULLY.

1. The **deadline** for this assignment is **19 March 2021, Friday, 1pm**. The submission folders will close shortly after 1pm. Late submission will not be accepted -- you will receive 0 mark.
2. Complete this assignment **on your own** without collaborating or discussing with anyone. If found to have cheated in this assignment, you will receive an F grade for this module as well as face other disciplinary sanctions. Take this warning seriously.
3. Please submit only **ONE pdf file** called **AxxxxxxxY.pdf**, where **AxxxxxxxY** is your student number. Marks will be deducted if you deposit multiple files in the submission folder or your submitted file is not a pdf file.
4. Please keep your submission **short**. You only need to submit your answers; you do not need to include the questions.
5. Answers may be typed or handwritten. In case of the latter, please ensure that you use **dark ink** and your handwriting is **neat and legible**. Marks may be deducted for untidy work or illegible handwriting.
6. Upload your file to LumiNUS > Files > Assignment 2 > (your tutorial group) > (your personal folder).
7. There are FIVE (5) questions on SIX (6) printed pages in this paper.
8. The questions are worth **40 marks** in total.

=== END OF INSTRUCTIONS ===

Question 0. Submission instructions (3 marks)

(a) You have named your file with your student number, i.e., **AxxxxxxxY.pdf**. (1 mark)

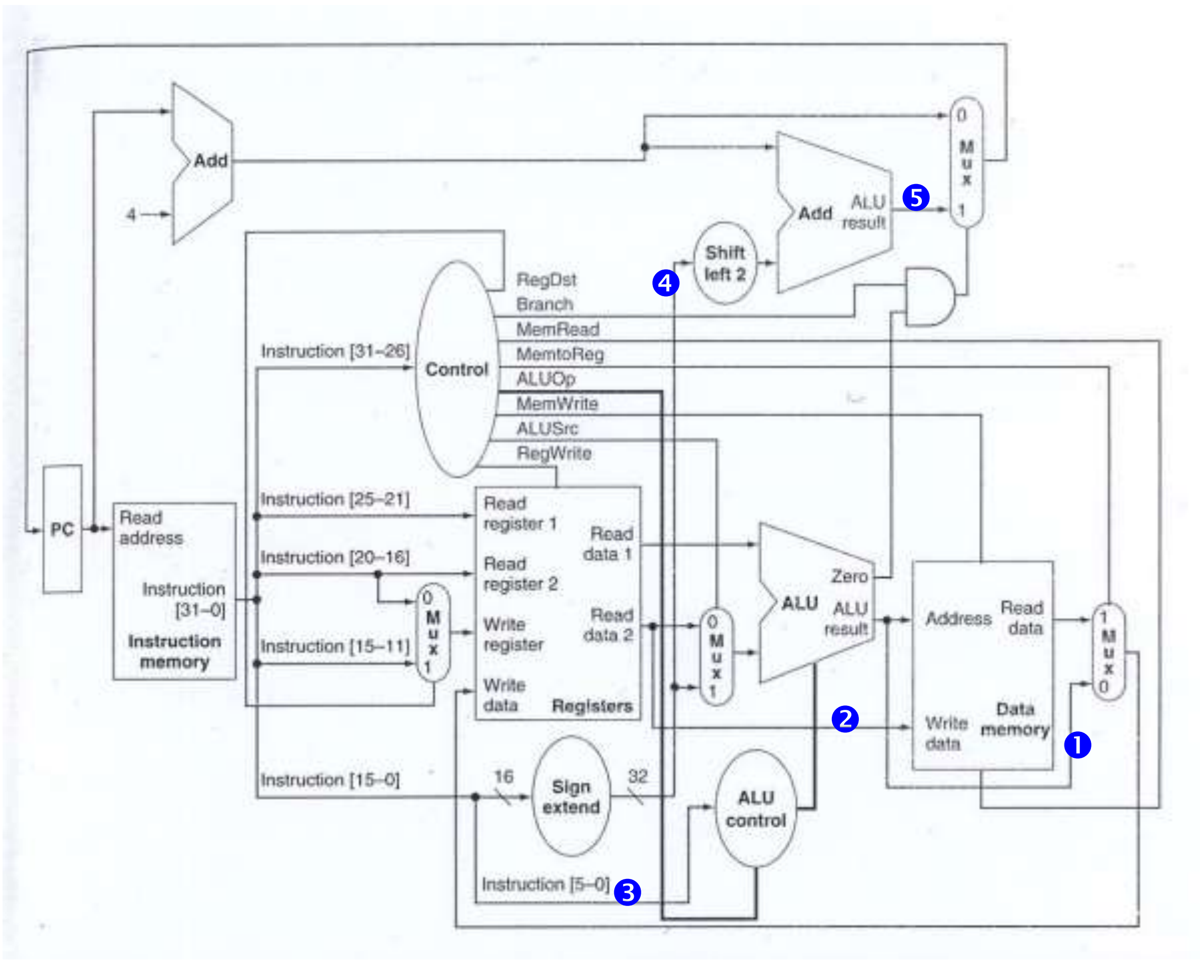
(b) You have submitted your assignment as a **single PDF file** and no multiple copies, i.e. there should only be ONE file in your submission folder. (1 mark)

(c) Your assignment submission has your **tutorial group number**, **student number** and **name** at the top of the first page of your file. (All three items must be present.) (1 mark)

Question 1. Datapath (8 marks)

[Past-year's exam question] Consider the MIPS datapath with the values of the registers shown below. All values are in hexadecimal.

```
R0  (r0) = 0x00000000 | R1  (at) = 0x00002000
R2  (v0) = 0x00000001 | R3  (v1) = 0x0000000a
R4  (a0) = 0x00000005 | R5  (a1) = 0x7ffff000
R6  (a2) = 0x7ffff004 | R7  (a3) = 0x000000b0
R8  (t0) = 0x00000001 | R9  (t1) = 0x00000c00
R10 (t2) = 0x0000c000 | R11 (t3) = 0xfffffff0
R12 (t4) = 0xf0000000 | R13 (t5) = 0x00000fff
R14 (t6) = 0x00006200 | R15 (t7) = 0x00000e00
R16 (s0) = 0x00300000 | R17 (s1) = 0x00000c00
R18 (s2) = 0x00040200 | R19 (s3) = 0x00011000
R20 (s4) = 0x00030200 | R21 (s5) = 0x10000000
R22 (s6) = 0x00055000 | R23 (s7) = 0xf0000000
R24 (t8) = 0x00000005 | R25 (t9) = 0x0000d000
R26 (k0) = 0x00000000 | R27 (k1) = 0x00000000
R28 (gp) = 0x10008000 | R29 (sp) = 0x7fffeff4
R30 (s8) = 0x1000000f | R31 (ra) = 0x00400018
```



Question 1. (continue…)

The current PC value is **200** and the instruction being executed is:

```
lw $t6, -24($sp)
```

Fill in the values of the fields in the table below. (8 marks)

For the 8th data row onwards in the table below, if your value is not in base ten, make sure you write the prefix (eg: 0x123) or subscript (eg: 123_{16}) to specify the base. Values without prefix or subscript will be treated as if they are in base ten by default.

Field	Value
RegDst	0
MemRead	1
Branch	0
MemtoReg	1
MemWrite	0
ALUSrc	1
RegWrite	1
Instruction[31-26]	35 or 0x23 or 0b100011
Instruction[25-21]	29 or 0x1D or 0b11101
Instruction[20-16]	14 or 0x0E or 0b01110
Instruction[15-11]	31 or 0x1F or 0b11111
❶	`0x7FFF EFDC`
❷	`0x0000 6200`
❸	`0x28` or `0b101000`
❹	`0xFFFF FFE8`
❺	108 or `0x6C`

Question 2. Simplification (12 marks)

(a) Simplify the following Boolean expression into its <u>simplest SOP expression</u>, showing each step with justification (i.e. citing the law/theorem used). Marks will be deducted if a step is not justified or wrongly justified. [4 marks]

$$A \cdot B' \cdot E \cdot L' \cdot O \cdot P' \cdot T' \cdot W \cdot H' + A \cdot L + A \cdot L' \cdot T'$$

You may skip obvious steps, such as $(A \cdot B')'$ straight to $(A'+B)$ [De Morgan's theorem] instead of first to $(A'+(B')')$ [De Morgan's] then to $(A'+B)$ [involution]; or $(A' \cdot B + A)$ straight to $(A+B)$ [absorption theorem 2] instead of first to $(A + A' \cdot B)$ [commutative law] then to $(A+B)$ [absorption theorem 2].

Recall that the AND operator · must be present. Writing AB instead of $A \cdot B$ will incur penalty.

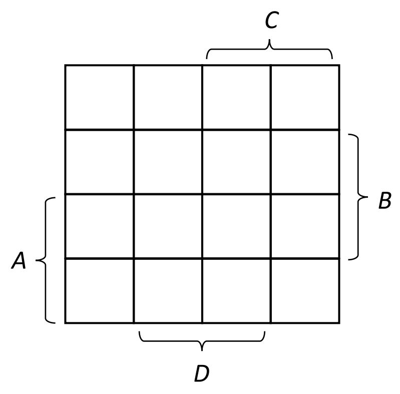
Answer:

$A \cdot B' \cdot E \cdot L' \cdot O \cdot P' \cdot T' \cdot W \cdot H' + A \cdot L + A \cdot L' \cdot T'$

Step 1:	$= \underline{A \cdot L' \cdot T'} \cdot B' \cdot E \cdot O \cdot P' \cdot W \cdot H' + \underline{A \cdot L' \cdot T'} + A \cdot L$	(associative law; commutative law)
Step 2:	$= A \cdot L' \cdot T' + A \cdot L$	(absorption theorem 1)
Step 3:	$= A \cdot (\underline{L' \cdot T' + L})$	(distributive law)
Step 4:	$= A \cdot (T' + L)$	(absorption theorem 2)
Step 5:	$= A \cdot T' + A \cdot L$	(distributive law)

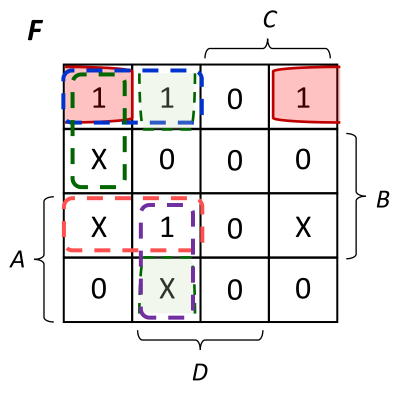
For each of the Boolean functions in parts (b) and (c) below, write (i) the number of PIs (prime implicants) in the K-map of the function, (ii) the number of EPIs (essential prime implicants) in the K-map of the function, (iii) its simplest SOP expression, and (iv) its simplest POS expression. If there are more than one simplest SOP/POS expression, you need to give only one.

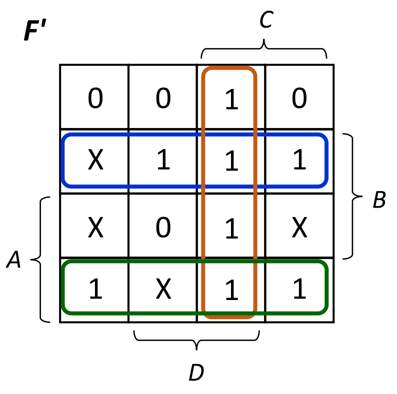
You may show your K-maps but they won't be graded. Your K-maps may be useful for us to locate the mistakes you made. If you are drawing the K-maps, please use the following layout. You may use X for don't cares.



(b) $F(A,B,C,D) = \Sigma m(0,1,2,13) + \Sigma X(4,9,12,14)$. [4 marks]

Answer:





PIs: 6 ($A' \cdot B' \cdot D'$, $A' \cdot B' \cdot C'$, $A' \cdot C' \cdot D'$, $B' \cdot C' \cdot D$, $A \cdot B \cdot C'$, $A \cdot C' \cdot D$.) (Only the number is needed.)

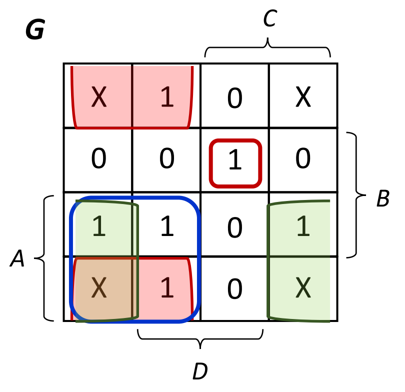
EPIs: 1 ($A' \cdot B' \cdot D'$.) (Only the number is needed.)

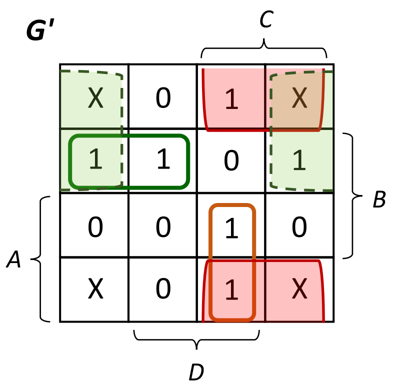
Simplified SOP expression: $F = A' \cdot B' \cdot D' + A' \cdot B' \cdot C' + A \cdot B \cdot C'$
or $A' \cdot B' \cdot D' + A' \cdot B' \cdot C' + A \cdot C' \cdot D$
or $A' \cdot B' \cdot D' + B' \cdot C' \cdot D + A \cdot B \cdot C'$
or $A' \cdot B' \cdot D' + B' \cdot C' \cdot D + A \cdot C' \cdot D$

Simplified POS expression: $F = (A+B') \cdot (C'+D') \cdot (A'+B)$ or $(A+B') \cdot (C'+D') \cdot (A'+D)$.
(Working $F' = A' \cdot B + C \cdot D + A \cdot B'$ or $F' = A' \cdot B + C \cdot D + A \cdot D'$.)

(c) $G(A,B,C,D) = \Pi M(3,4,5,6,11,15) \cdot \Pi X(0,2,8,10)$. [4 marks]

Answer:





PIs: 4 ($A \cdot C'$, $B' \cdot C'$, $A \cdot D'$, $A' \cdot B \cdot C \cdot D$.) (Only the number is needed.)

EPIs: 4 ($A \cdot C'$, $B' \cdot C'$, $A \cdot D'$, $A' \cdot B \cdot C \cdot D$.) (Only the number is needed.)

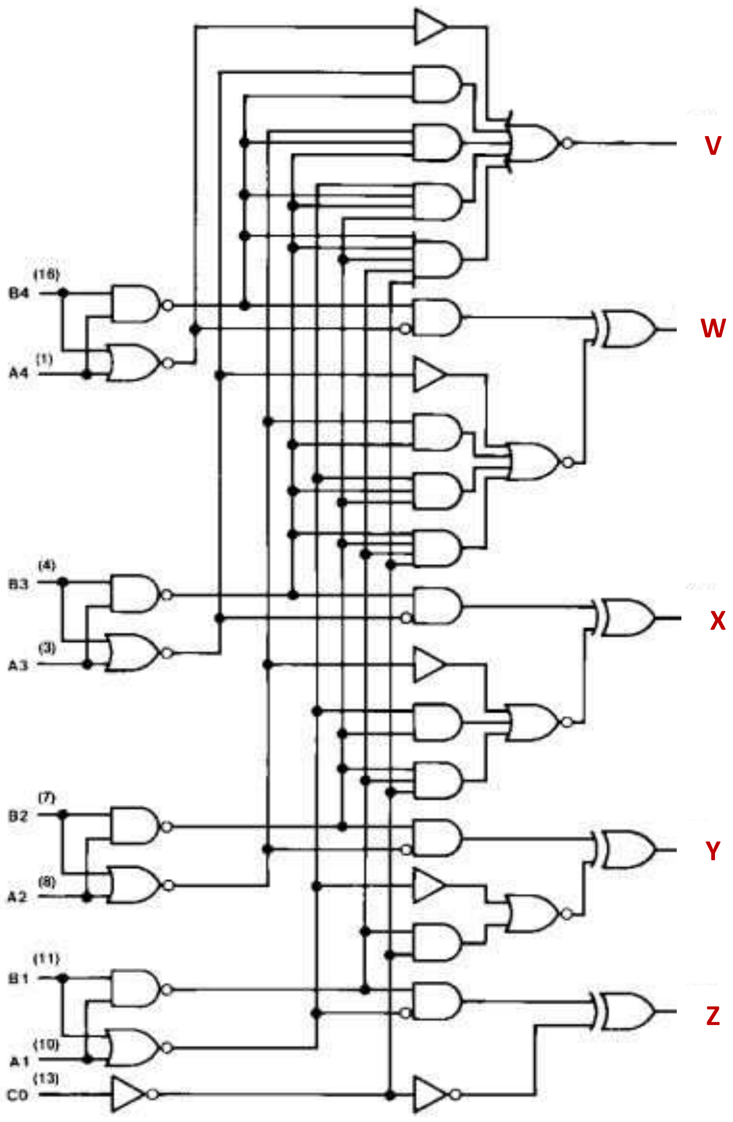
Simplified SOP expression: $G = A \cdot C' + B' \cdot C' + A \cdot D' + A' \cdot B \cdot C \cdot D$

Simplified POS expression: $G = (B+C') \cdot (A+D) \cdot (A+B'+C) \cdot (A'+C'+D')$

(Working $G' = B' \cdot C + A' \cdot D' + A' \cdot B \cdot C' + A \cdot C \cdot D$.)

Question 3. Circuit Analysis (4 marks)

The following circuit diagram is taken from the data sheet of an IC chip. Identify the circuit below, which takes in C_0, A_1, A_2, A_3, A_4, B_1, B_2, B_3 and B_4 as inputs and generates V, W, X, Y and Z as the outputs. Give the **name** of this circuit and **describ**e what it does by relating the outputs with the inputs.



Answer:

This is a 4-bit parallel adder/4-bit full adder/4-bit look-ahead carry adder that takes in C_0 and two 4-bit binary numbers $A_4A_3A_2A_1$ and $B_4B_3B_2B_1$ and generates the output VWXYZ = C_0 + $A_4A_3A_2A_1$ + $B_4B_3B_2B_1$.

Note: The inputs $A_4A_3A_2A_1$ and $B_4B_3B_2B_1$ must be explicitly mentioned, not merely A and B which do not appear in the logic diagram.

Question 4. Circuit Design (7 marks)

A combinational circuit takes in a 5-bit input *ABCDE* which is a value in excess 8 representation, and generates an output *P* such that *P*=1 if *ABCDE* represents a prime number, or *P*=0 otherwise.

For example, if *ABCDE*=10001, then *P*=0; if *ABCDE*=11001, then *P*=1.

You may assume that negative values will not be supplied to this circuit.

(a) Draw a 5-variable K-map for *P* (draw one K-map with *A*=0 and another with *A*=1). [2 marks]

(b) Write out the simplified SOP expression for *P*. [3 marks]

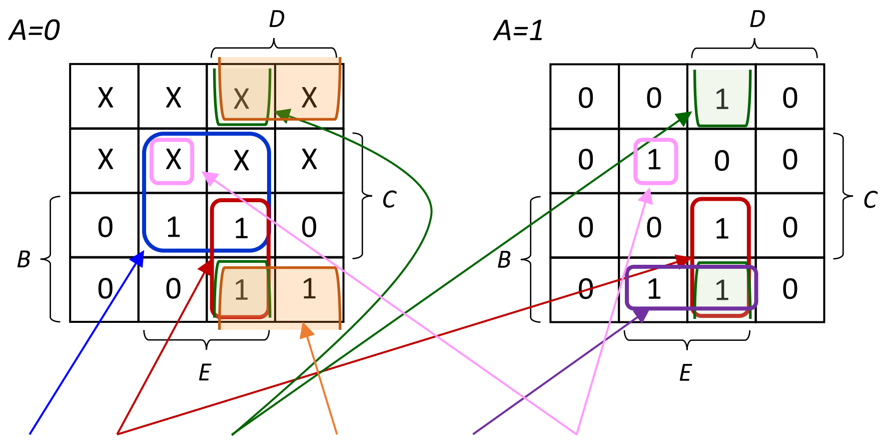
(c) Let *V*(*A*,*B*,*C*,*D*,*E*) be the validity function, that is, *V* returns 1 if the input is valid, or 0 otherwise. Implement *V* using the fewest number of gates. Draw your circuit. [2 marks]

Answer:

A	*B*	*C*	*D*	*E*	*P*
0	0	0	0	0	X
0	0	0	0	1	X
0	0	0	1	0	X
0	0	0	1	1	X
0	0	1	0	0	X
0	0	1	0	1	X
0	0	1	1	0	X
0	0	1	1	1	X
0	1	0	0	0	0
0	1	0	0	1	0
0	1	0	1	0	1
0	1	0	1	1	1
0	1	1	0	0	0
0	1	1	0	1	1
0	1	1	1	0	0
0	1	1	1	1	1

A	*B*	*C*	*D*	*E*	*P*
1	0	0	0	0	0
1	0	0	0	1	0
1	0	0	1	0	0
1	0	0	1	1	1
1	0	1	0	0	0
1	0	1	0	1	1
1	0	1	1	0	0
1	0	1	1	1	0
1	1	0	0	0	0
1	1	0	0	1	1
1	1	0	1	0	0
1	1	0	1	1	1
1	1	1	0	0	0
1	1	1	0	1	0
1	1	1	1	0	0
1	1	1	1	1	1

(a)

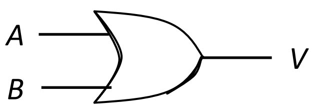


X	X	X	X
X	X	X	X
0	1	1	0
0	0	1	1

0	0	1	0
0	1	0	0
0	0	1	0
0	1	1	0

(b) $P = A'{\cdot}C{\cdot}E + B{\cdot}D{\cdot}E + C'{\cdot}D{\cdot}E + A'{\cdot}C'{\cdot}D + A{\cdot}B{\cdot}C'{\cdot}E + B'{\cdot}C{\cdot}D'{\cdot}E$

(c) $V = A + B$



Question 5. Block-level design (6 marks)

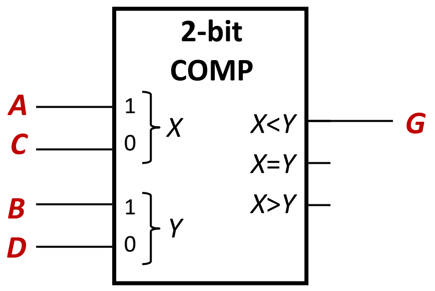
Note:

- Boolean constants (0 and 1) are always available;
- Complemented literals are not available.

(a) Implement the following Boolean function using a single 2-bit magnitude comparator with no additional logic gates.

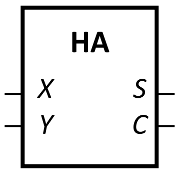
$$G(A,B,C,D) = \Sigma m(1, 4, 5, 6, 7, 13)$$

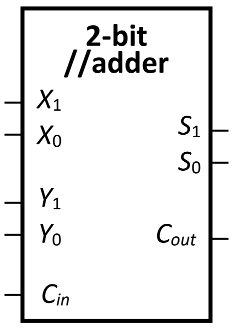
The block diagram of a 2-bit magnitude comparator is shown below. [3 marks]



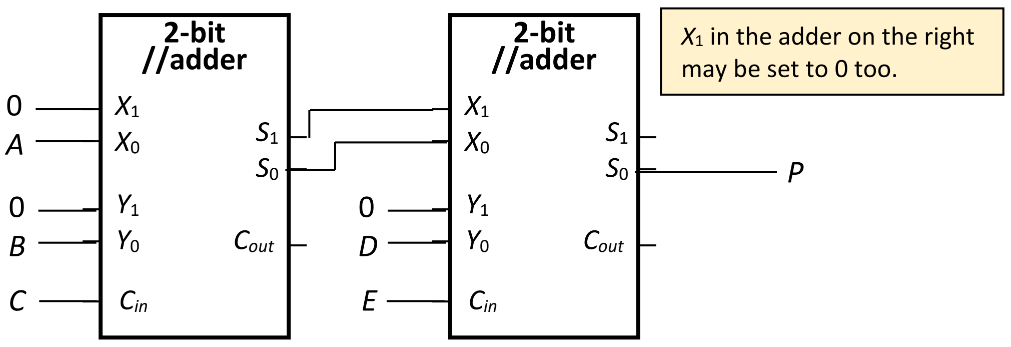
(b) A circuit takes in a 5-bit code *ABCDE* and generates *P*, the parity bit for the code, assuming even parity scheme, which means that the total number of 1s in *ABCDE* and *P* must be even. For example, if *ABCDE* = 00110, then *P*=0; if *ABCDE* = 10101, then *P*=1.

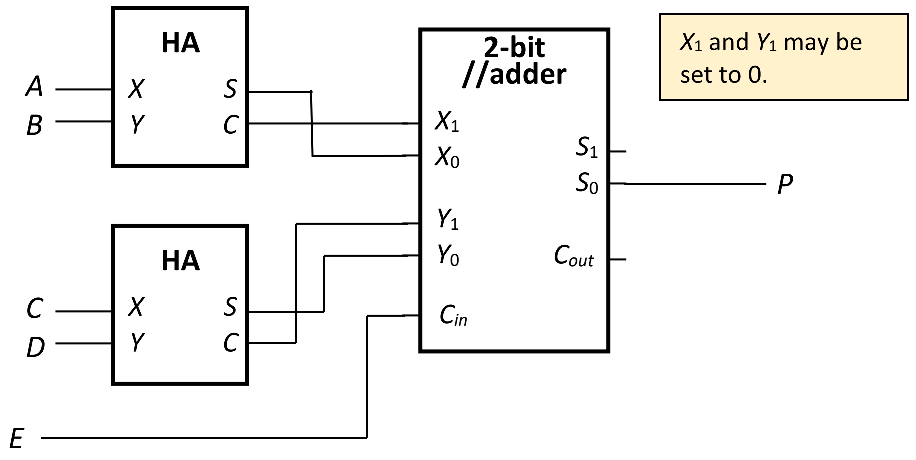
Implement the above circuit using the fewest number of half adders (HA) and 2-bit parallel adders, and no other logic gates. The block diagrams for a half adder and a 2-bit parallel adder are shown below. You are not allowed to change the block diagrams. [3 marks]





Possible answers (there could be others):





=== END OF PAPER ===