Huddle Board Exercise for Module 4 – No. 1a
Monday, April 14, 2014

Complete the following ABEL file so that it implements a 4-bit *carry look-ahead* adder based on the DECLARATIONS section provided below.

```
MODULE cla4
TITLE '4-bit Carry Look-Ahead Adder'

DECLARATIONS

X0..X3, Y0..Y3 pin;  " operands
CIN pin;  " carry in
S0..S3 pin istype 'com';  " sum outputs
C0..C3 pin istype 'com';  " carry outputs (C3 is carry out)
P0..P3 pin istype 'com';  " propagate functions

G0 = X0&Y0;  " generate functions
G1 = X1&Y1;
G2 = X2&Y2;
G3 = X3&Y3;

EQUATIONS

P0 = X0$Y0;   " propagate functions
P1 = X1$Y1;
P2 = X2$Y2;
P3 = X3$Y3;

" carry functions

C0 =     G0 # CIN&P0                                                    ;

C1 =     G1 # G0&P1 # CIN&P0&P1                                         ;

C2 =     G2 # G1&P2 # G0&P1&P2 # CIN&P0&P1&P2

                                                                        ;

C3 =     G3 # G2&P3 # G1&P2&P3 # G0&P1&P2&P3 # CIN&P0&P1&P2&P3

                                                                        ;

S0 = CIN$P0;  " sum functions
S1 = C0$P1;
S2 = C1$P2;
S3 = C2$P3;

END
```

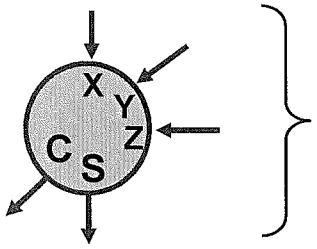
```
      C2  C1  C0  CIN
      X3  X2  X1  X0
  +   Y3  Y2  Y1  Y0
  ---------------------
  C3  S3  S2  S1  S0
```

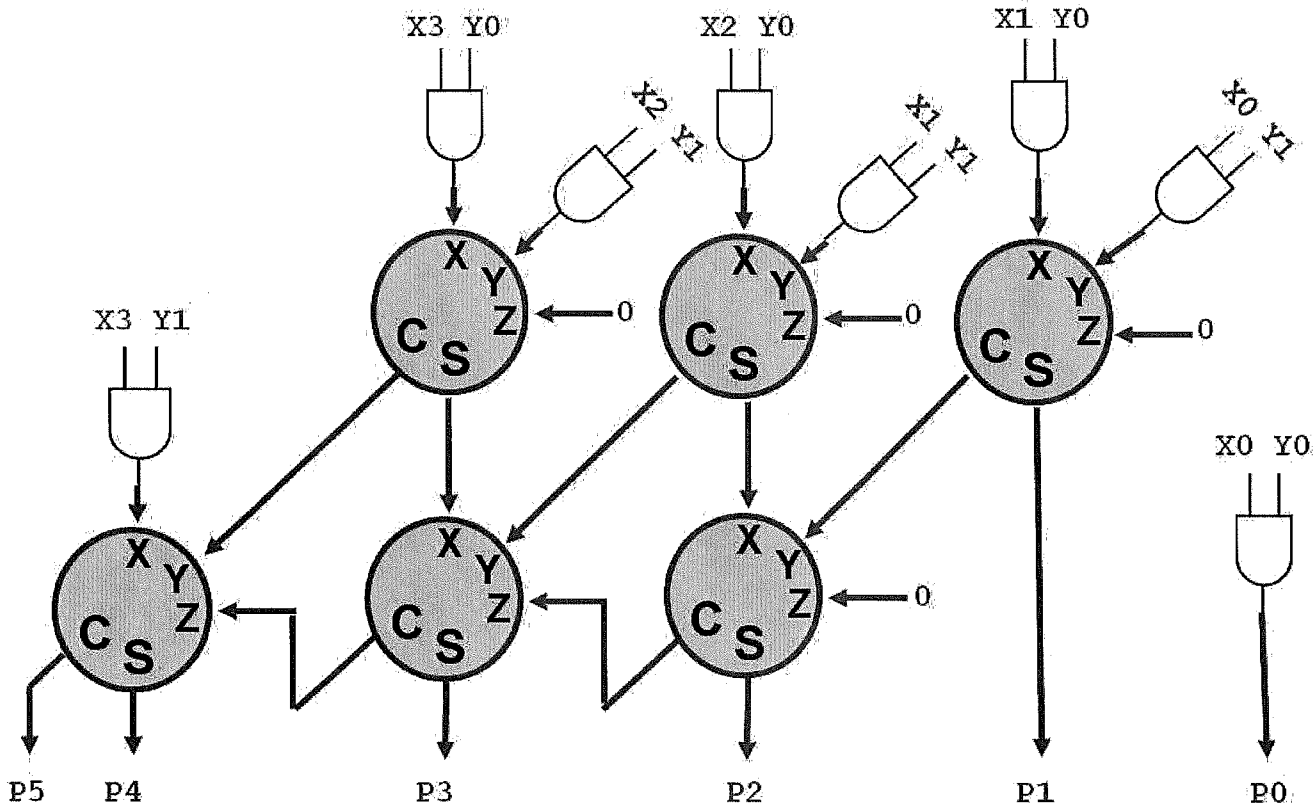
Huddle Board Exercise for Module 4 – No. 1b
Monday, April 14, 2014

Draw a circuit that multiplies a 4-bit unsigned binary number X3 X2 X1 X0 by a 2-bit unsigned binary number Y1 Y0, using an array of full-adder cells. Determine the worst case propagation delay if each full adder takes 10 ns to produce its C and S outputs, and each AND gate (used to generate the product components) has 5 ns of propagation delay.

```
          X3    X2    X1    X0
                ×     Y1    Y0
          ------------------------
          X3Y0  X2Y0  X1Y0  X0Y0
+   X3Y1  X2Y1  X1Y1  X0Y1
-------------------------------------

P5    P4    P3    P2    P1    P0
```





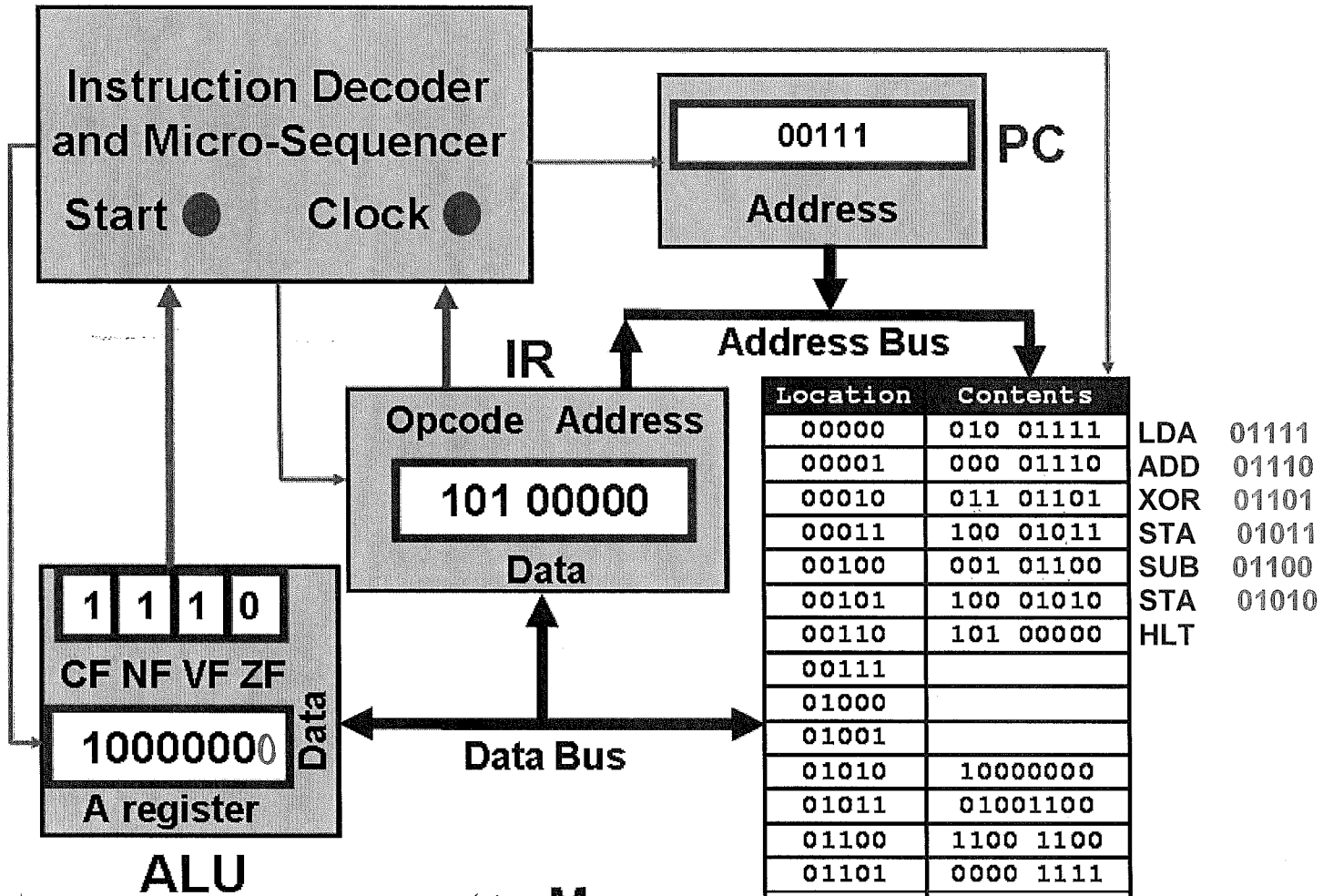
The worst case propagation delay is: ____45 ns_____.

Huddle Board Exercise for Module 4 – No. 2a
Wednesday, April 16, 2014

Assume the simple computer instruction set has been changed to the following:

Opcode	Mnemonic	Function Performed
0 0 0	ADD *addr*	Add contents of *addr* to contents of A
0 0 1	SUB *addr*	Subtract contents of *addr* from contents of A
0 1 0	LDA *addr*	Load A with contents of location *addr*
0 1 1	XOR *addr*	XOR contents of *addr* with contents of A
1 0 0	STA *addr*	Store contents of A at location *addr*
1 0 1	HLT	Halt – Stop, discontinue execution

On the instruction trace worksheet, below, show the *final result* of executing the program stored in memory *up to and including* the HLT instruction.



Memory

Huddle Board Exercise for Module 4 – No. 2b
Wednesday, April 16, 2014

From the **BLOCK DIAGRAM** for **one bit** ("i") of the ALU, complete the table below:

AOE	ALE	ALX	ALY	Function Performed	CF	ZF	NF	VF
0	1	0	0	LDA: [Q3..Q0] ← [D3..D0]	•	X	X	•
0	1	0	1	AND: [Q3..Q0] ← [Q3..Q0] ∩ [D3..D0]	•	X	X	•
0	1	1	0	SUB: [Q3..Q0] ← [Q3..Q0] – [D3..D0]	X	X	X	X
0	1	1	1	ADD: [Q3..Q0] ← [Q3..Q0] + [D3..D0]	X	X	X	X
1	0	d	d	OUT: [D3..D0] ← [Q3..Q0]	•	•	•	•
0	0	d	d	(no operation – retain state)	•	•	•	•

"X" indicates the flag is affected by the function performed, "•" indicates the flag is not affected