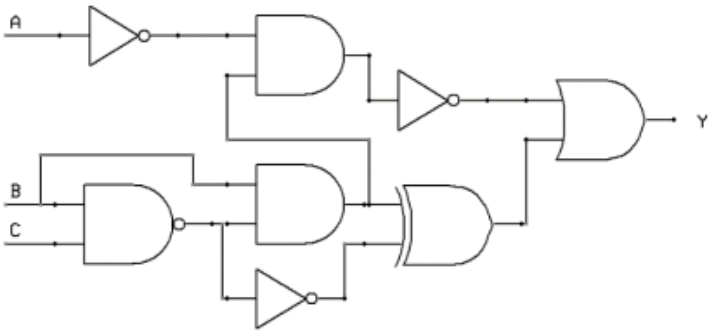
1. A. Write the Boolean expression of the following circuit diagram [2 marks].



$$Y = \overline{\left(\overline{A}.\left(B.\overline{CB}\right)\right)} + B.\overline{CB} \bigoplus \overline{\left(\overline{CB}\right)}$$

Or Simplified:

K = $\overline{A}\ .\ L$

L = $B\ .\ M$

M = $\overline{CB}$

N = $\overline{M}$

O = $\overline{K}$

P = $L \oplus N$

$$Y = P + O$$

B. Set up the truth table [9 marks]

A	B	C	$\overline{A}$	$\overline{A}\ .\ \left(B.\overline{CB}\right)$	$B.\overline{CB}$	$\overline{CB}$	$\overline{\left(\overline{CB}\right)}$	$\overline{\left(\overline{A}.\left(B.\overline{CB}\right)\right)}$	$B.\overline{CB} \bigoplus \overline{\left(\overline{CB}\right)}$	Y
0	0	0	1	0	0	1	0	1	0	1
0	0	1	1	0	0	1	0	1	0	1
0	1	0	1	1	1	1	0	0	1	1
0	1	1	1	0	0	0	1	1	1	1
1	0	0	0	0	0	1	0	1	0	1
1	0	1	0	0	0	1	0	1	0	1
1	1	0	0	0	1	1	0	1	1	1
1	1	1	0	0	0	0	1	1	1	1

A	B	C	$\overline{A}$	K	L	M	N	O	P	Y
0	0	0	1	0	0	1	0	1	0	1
0	0	1	1	0	0	1	0	1	0	1

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

2. A. Given the two following decimal numbers : 37, -126

i) Represent the two numbers in One's Complement representation (using 8-bit binary in the result) [4 marks]

37 as a binary number = 0010 0101

We find 126 as a binary number which = 0111 1110

To get ones complement, you invert the bits of the binary number:

37 represented in one's complement = 1101 1010

-126 represented in one's complement = 1000 0001

ii) Represent the two numbers in Two's Complement representation (using 8-bit binary in the result) [6 marks]

37 as a binary number = 0010 0101

We find 126 as a binary number which = 0111 1110

To get two's complement, you invert the bits of the binary number and add one.

37 inverted or represented in one's complement = 1101 1010

-126 inverted or represented in one's complement = 1000 0001

-126 represented in two's complement = 1000 0010

37 represented in two's complement = 1101 1011

B. Convert the positive number N=1010000001001 in single precision floating point format [3 marks]

Step 1: 1010000001001 = 1.010000001001 x 2^{12}

Step 2: N = $(-1)^{S}$ (1+F)(2^{E-127})

S = 0 (Positive Number)

F = 01000000100100000000000

E - 127 = 12, E = 139_{10} = 1000 1011_2

N =

0	10001011	01000000100100000000000

3. If main memory is of 4 giga bytes and every word is of 2 bytes how many bits do we need to address any single word in memory? [4 marks]

4 giga bytes = 4000mb, which is $2^{12} + 2^{20} = 2^{32}$

Memsize = number.words x word.size so,

4000mb = 2^{31} x 2^1

We need $\log_2 2^{31}$ or 31 bits to address each word

4. How much RAM memory can a 16-bit CPU can use? Provide your answer in bytes. [4 marks]

A 16 bit cpu means that the address space capacity = 2^{16} words

2^{16} words = 64 x 2^{10} = 64KB = 65,536 bytes

So the maximum amount of RAM that it can use is 65,536 bytes.

5. Consider a 7-stage pipelined CPU where every stage is 30nsecs. How much time does it take to execute 1000 CPU instructions if no stall cycles occur? Provide the answer in nsecs. [5 marks]

7 x 30 = 210

1000 x 210 = 210,000nsecs

6. Let a CPU with CPU frequency 2 GHz. It executed 1.5 million instructions in 6 million CPU clock cycles.

A. What is the average value of the CPI for this CPU? [4 marks]

6,000,000 / 1,500,000 = 4

The average value of the CPI for this CPU = 4

B. How long did it take to complete all the instructions? [3 marks]

6,000,000 / 2GHz = 2billion = 0.003 seconds

7. A CPU has an average CPI of 2.5. It took 1.6 seconds to execute 9.6 million instructions. What is the speed of this CPU? [6 marks]

Speed = instruction / (time * CPI)

Speed = 9.6 million / 1.6 * 2.5

9.6 million / 4 = 2.4 million = 2.4MHz

8. Convert the following C code into assembly code. Do not simplify the code. The assembly code must be a) provided as a separate .asm file and b) included in the delivered .docx file [50 marks]

```
; A simple template for assembly programs.

.386  ; Specify instruction set

.model flat, stdcall  ; Flat memory model, std. calling convention

.stack 4096 ; Reserve stack space

ExitProcess PROTO, dwExitCode: DWORD  ; Exit process prototype


.data ; data segment

; define your variables here

A BYTE 3,2,3,1,7,5,0,8,9,2

B BYTE 0,0,0,0,0,0,0,0,0,0


.code ; code segment


main PROC ; main procedure

; write your assembly code here

lea ebx, A ; loads A into ebx

mov esi, 0


update:

; (3*i + 1)/5
```

```
mov eax, 3 ; moves 3 into eax
mul esi ; multiplies i by eax
add eax, 1 ; adds 1 to eax
mov ecx, 5 ; moves 5 into ecx
div ecx ; divs eax by 5
mov ecx, eax ; moves eax into ecx

; (2*i + ecx)
mov eax, 2 ; moves 2 into eax
mul esi ;multiplies i bys esi
add eax, ecx ; adds ecx to eax

; 2*eax
mov ecx, 2 ; moves 2 into ecx
mul ecx ; multiplies 2 by eax

;B[i] = A[i] + eax
mov ecx, [ebx + TYPE A * esi] ; moves i position of array into ecx
add eax, ecx ; adds ecx to eax
mov [ebx + TYPE B * esi], eax ; moves eax into the i position of B array in ebx


inc esi;
cmp esi, 10
jne update ; end or start of loop


INVOKE ExitProcess, 0 ; call exit function
```

```
main ENDP ; exit main procedure

END main  ; stop assembling
```