# Microprocessor

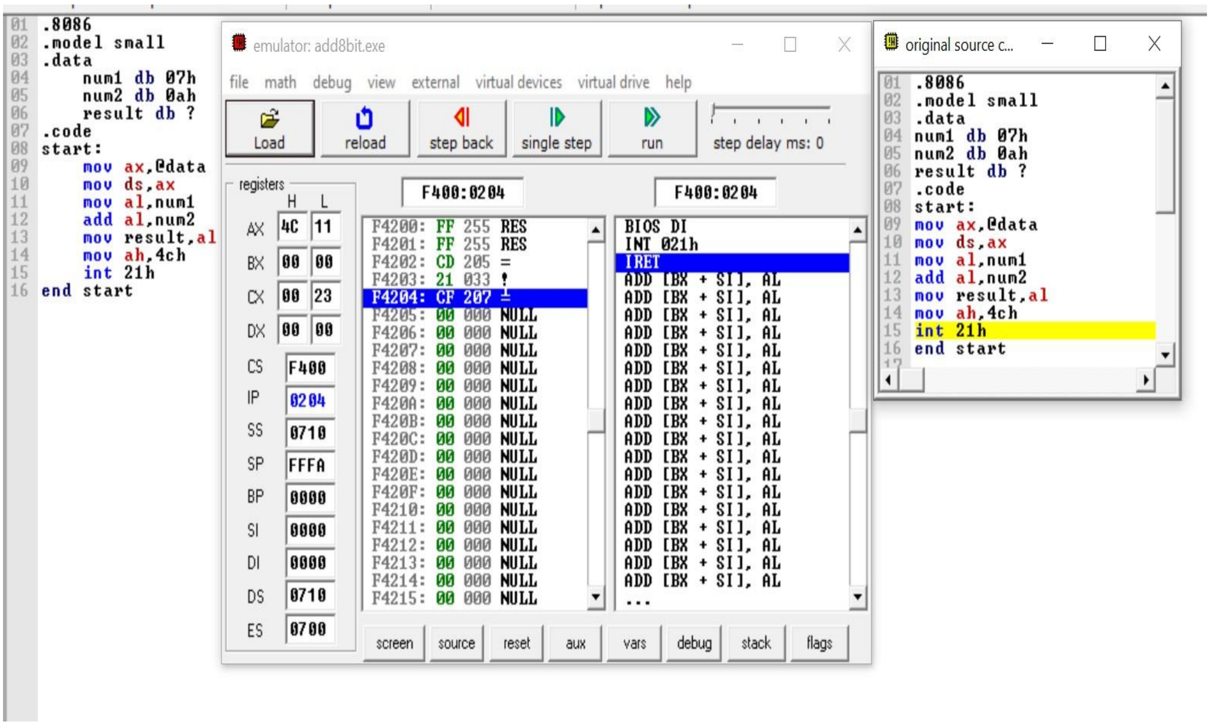
# Experiment 1

Name – Warren Fernandes

Roll No – 8940

## 1) 8 bit Addition

```
.8086
.model small
.data
num1 db 07h
num2 db 0ah
result db ?
.code
start:
mov ax,@data
mov ds,ax
mov al,num1
add al,num2
mov result,al
mov ah,4ch
int 21h
end start
```



```
.8086
.model small
.data
     num1 db 07h
     num2 db 0ah
     result db ?
.code
start:
     mov ax,@data
     mov ds,ax
     mov al,num1
     add al,num2
     mov result,al
     mov ah,4ch
     int 21h
end start
```

emulator: add8bit.exe

registers

|  | H | L |
|---|---|---|
| AX | 4C | 11 |
| BX | 00 | 00 |
| CX | 00 | 23 |
| DX | 00 | 00 |
| CS | F400 | |
| IP | 0204 | |
| SS | 0710 | |
| SP | FFFA | |
| BP | 0000 | |
| SI | 0000 | |
| DI | 0000 | |
| DS | 0710 | |
| ES | 0700 | |

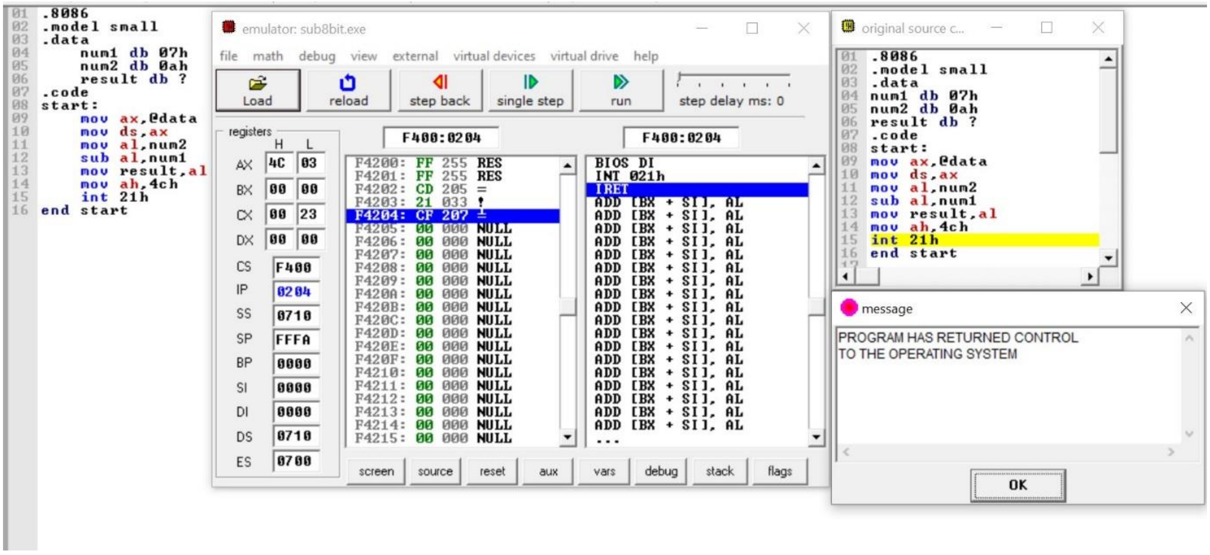
```
F400:0204

F4200: FF 255 RES
F4201: FF 255 RES
F4202: CD 205 =
F4203: 21 033 !
F4204: CF 207 ╧
F4205: 00 000 NULL
F4206: 00 000 NULL
F4207: 00 000 NULL
F4208: 00 000 NULL
F4209: 00 000 NULL
F420A: 00 000 NULL
F420B: 00 000 NULL
F420C: 00 000 NULL
F420D: 00 000 NULL
F420E: 00 000 NULL
F420F: 00 000 NULL
F4210: 00 000 NULL
F4211: 00 000 NULL
F4212: 00 000 NULL
F4213: 00 000 NULL
F4214: 00 000 NULL
F4215: 00 000 NULL
```

```
F400:0204

BIOS DI
INT 021h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

```
.8086
.model small
.data
num1 db 07h
num2 db 0ah
result db ?
.code
start:
mov ax,@data
mov ds,ax
mov al,num1
add al,num2
mov result,al
mov ah,4ch
int 21h
end start
```

## 2) 8 bit Subtraction

```
.8086
.model small
.data
num1 db 07h
num2 db 0ah
result db ?
.code
start:
mov ax,@data
mov ds,ax
mov al,num2
sub al,num1
mov result,al
mov ah,4ch
int 21h
end start
```



```
.8086
.model small
.data
     num1 db 07h
     num2 db 0ah
     result db ?
.code
start:
     mov ax,@data
     mov ds,ax
     mov al,num2
     sub al,num1
     mov result,al
     mov ah,4ch
     int 21h
end start
```

```
    H  L
AX  4C 03
BX  00 00
CX  00 23
DX  00 00
CS  F400
IP  0204
SS  0710
SP  FFFA
BP  0000
SI  0000
DI  0000
DS  0710
ES  0700
```

```
F400:0204

F4200: FF 255 RES
F4201: FF 255 RES
F4202: CD 205 =
F4203: 21 033 !
F4204: CF 207 ±
F4205: 00 000 NULL
F4206: 00 000 NULL
F4207: 00 000 NULL
F4208: 00 000 NULL
F4209: 00 000 NULL
F420A: 00 000 NULL
F420B: 00 000 NULL
F420C: 00 000 NULL
F420D: 00 000 NULL
F420E: 00 000 NULL
F420F: 00 000 NULL
F4210: 00 000 NULL
F4211: 00 000 NULL
F4212: 00 000 NULL
F4213: 00 000 NULL
F4214: 00 000 NULL
F4215: 00 000 NULL
```

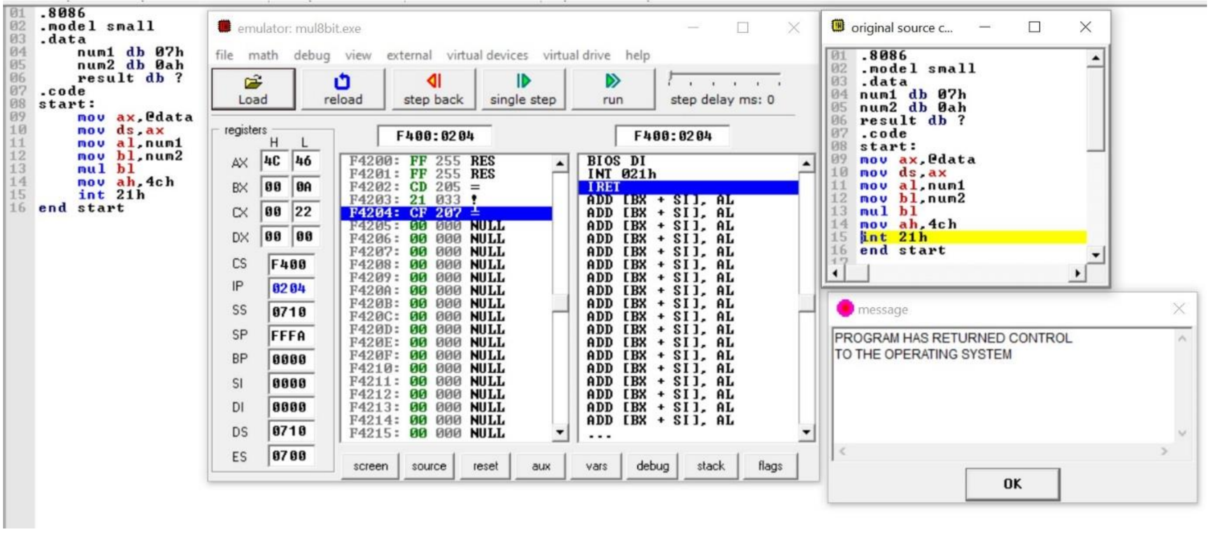
```
F400:0204

BIOS DI
INT 021h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

```
.8086
.model small
.data
num1 db 07h
num2 db 0ah
result db ?
.code
start:
mov ax,@data
mov ds,ax
mov al,num2
sub al,num1
mov result,al
mov ah,4ch
int 21h
end start
```

PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM

## 3) 8 bit Multiplication

```
.8086
.model small
.data
num1 db 07h
num2 db 0ah
result db ?
.code
start:
mov ax,@data
mov ds,ax
mov al,num1
mov bl,num2
mul bl
mov ah,4ch
int 21h
end start
```



```
.8086
.model small
.data
      num1  db  07h
      num2  db  0ah
      result  db  ?
.code
start:
      mov  ax,@data
      mov  ds,ax
      mov  al,num1
      mov  bl,num2
      mul  bl
      mov  ah,4ch
      int  21h
end  start
```

emulator: mul8bit.exe

| | H | L |
|---|---|---|
| AX | 4C | 46 |
| BX | 00 | 0A |
| CX | 00 | 22 |
| DX | 00 | 00 |
| CS | F400 | |
| IP | 0204 | |
| SS | 0710 | |
| SP | FFFA | |
| BP | 0000 | |
| SI | 0000 | |
| DI | 0000 | |
| DS | 0710 | |
| ES | 0700 | |

```
F400:0204
F4200:  FF  255  RES
F4201:  FF  255  RES
F4202:  CD  205  =
F4203:  21  033  !
F4204:  CF  207  ±
F4205:  00  000  NULL
F4206:  00  000  NULL
F4207:  00  000  NULL
F4208:  00  000  NULL
F4209:  00  000  NULL
F420A:  00  000  NULL
F420B:  00  000  NULL
F420C:  00  000  NULL
F420D:  00  000  NULL
F420E:  00  000  NULL
F420F:  00  000  NULL
F4210:  00  000  NULL
F4211:  00  000  NULL
F4212:  00  000  NULL
F4213:  00  000  NULL
F4214:  00  000  NULL
F4215:  00  000  NULL
```

```
F400:0204
BIOS DI
INT 021h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

original source c...

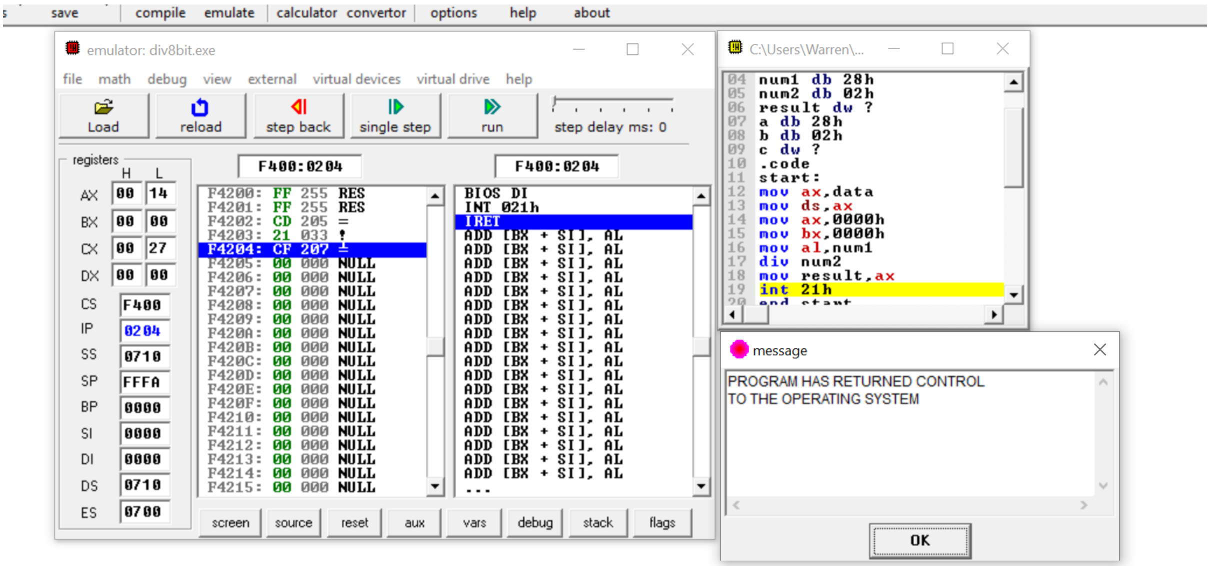
```
.8086
.model small
.data
num1 db 07h
num2 db 0ah
result db ?
.code
start:
mov ax,@data
mov ds,ax
mov al,num1
mov bl,num2
mul bl
mov ah,4ch
int 21h
end start
```

message

PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM

## 4) 8 bit Division

```
.8086
.model small
.data
num1 db 28h
num2 db 02h
result db ?
.code
start:
mov ax,data
mov ds,ax
mov ax,00h
mov bx,00h
mov al,num1
div num2
mov result,ax
int 21h
end start
```



```
registers
     H    L
AX   00   14
BX   00   00
CX   00   27
DX   00   00
CS   F400
IP   0204
SS   0710
SP   FFFA
BP   0000
SI   0000
DI   0000
DS   0710
ES   0700
```

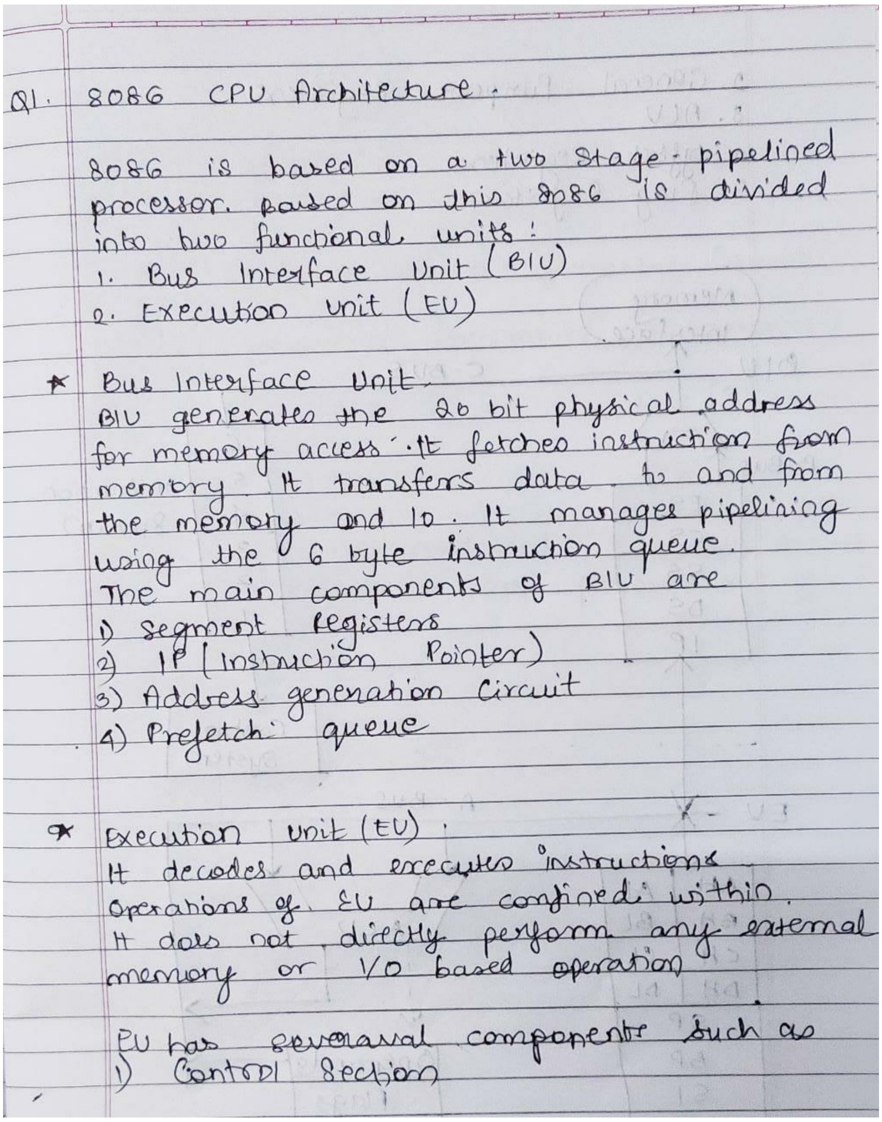
```
F400:0204

F4200:  FF  255  RES
F4201:  FF  255  RES
F4202:  CD  205  =
F4203:  21  033  !
F4204:  CF  207  ╧
F4205:  00  000  NULL
F4206:  00  000  NULL
F4207:  00  000  NULL
F4208:  00  000  NULL
F4209:  00  000  NULL
F420A:  00  000  NULL
F420B:  00  000  NULL
F420C:  00  000  NULL
F420D:  00  000  NULL
F420E:  00  000  NULL
F420F:  00  000  NULL
F4210:  00  000  NULL
F4211:  00  000  NULL
F4212:  00  000  NULL
F4213:  00  000  NULL
F4214:  00  000  NULL
F4215:  00  000  NULL
```

```
F400:0204

BIOS DI
INT 021h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

```
num1 db 28h
num2 db 02h
result dw ?
a db 28h
b db 02h
c dw ?
.code
start:
mov ax,data
mov ds,ax
mov ax,0000h
mov bx,0000h
mov al,num1
div num2
mov result,ax
int 21h
end start
```

PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM

## POSTLAB:



Q1. 8086 CPU Architecture :

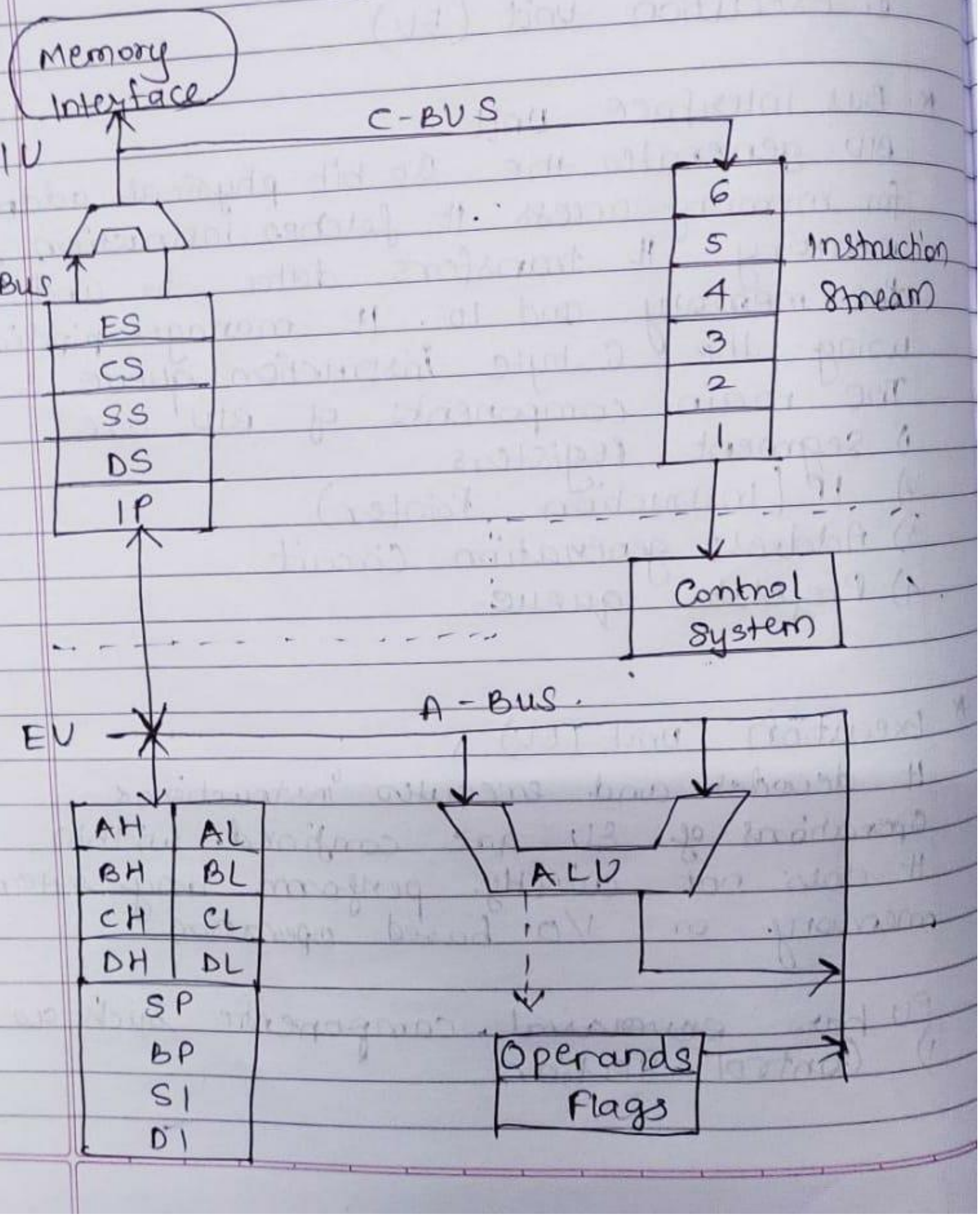
8086 is based on a two stage pipelined processor. Based on this 8086 is divided into two functional units :
1. Bus Interface Unit (BIU)
2. Execution Unit (EU)

★ Bus Interface Unit.
BIU generates the 20 bit physical address for memory access. It fetches instruction from memory. It transfers data to and from the memory and IO. It manages pipelining using the 6 byte instruction queue.
The main components of BIU are
1) Segment Registers
2) IP (Instruction Pointer)
3) Address generation Circuit
4) Prefetch queue

★ Execution Unit (EU) :
It decodes and executes instructions
Operations of EU are confined within.
It does not directly perform any external memory or I/O based operation

EU has severalaral components such as
1) Control Section

2. General Purpose Registers
3. ALU
4. Offset Registers
5. Flag Registers.



Q2. Syntax for Instruction format

Assembly language statements are entered one statement per line.

Each Statement follows the following format.

[label] mnemonic [operands] [; comment]

The fields in the square brackets [] are optional.
A basic instruction has two parts the first one is the name of the instruction which is to be excecuted, and the second are the operands or the parameters of the command

For eg;

```
inc count.
Add ah, bh
```