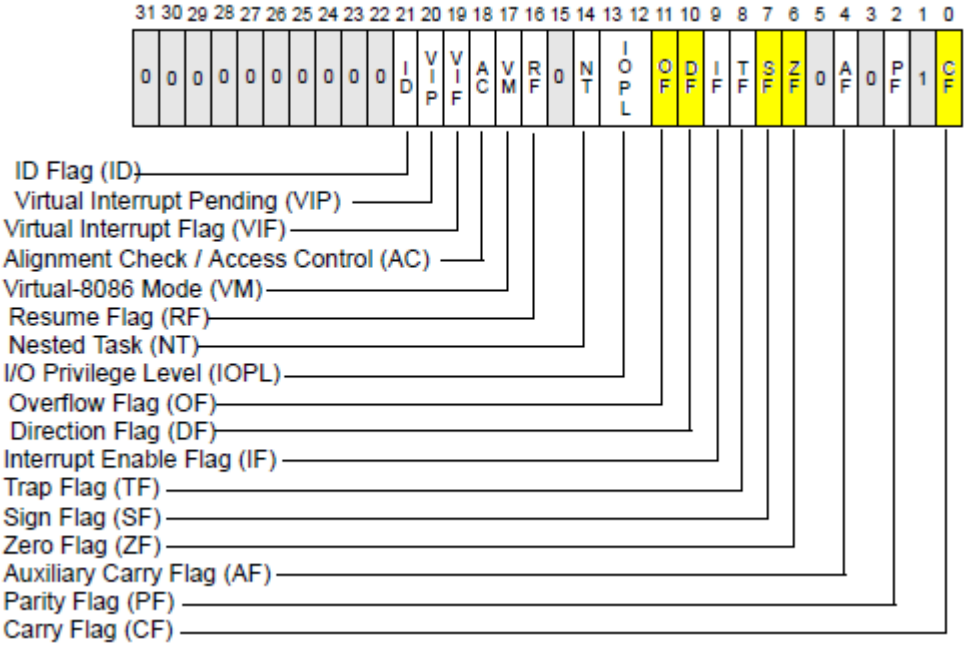
(100 pts) CS3843 Computer Organization Exam #3 Name/abc123:_____
(50 pts) Part 2

Coding: (50 pts)

1. (15 pts) The EFlags register in x86 has bit assignments for the primary flags as shown below. Assume the value of the EFlags register has been put in eax (shown how by code below), write the assembly code to effect the following changes to the flags: Set CF, clear SF, clear DF, toggle ZF, toggle OF. Do not use built-in assembly instructions such as "cld".

```
// Puts EFlags register on the stack
pushf        ; save eflags register to stack
pop eax      ; put contents of eflags register, from stack, into eax
```



// Code to alter the flags here: Leave all other flags unchanged.

// Puts the altered flags back into the EFlags register

```
push eax     ; put altered eax on to the stack
popf         ; put values into flags register
```

2. (10 pts) Write the assembly instructions to set up a function's stack frame, reserve 24 bytes for local variables, and store the second parameter in one of those local variables.

3. (10 pts) Convert the following C program to assembly:

```
int x, y;
char text[] = “cool cat”  // make room for the array on the stack, but no need
                          // to write the code to populate it
y = 0;
for( x = 0; x < 10; x++) {
   if( text[x] == ‘c’) y = y + 1;   // lowercase ‘c’ == 0x63 in hex
}
```

4. (10 pts) Convert the following assembly program to a C function. Show the function call as well.

```
55          push  ebp
8b ec       mov   ebp, esp
51          push  ecx

8b 45 0c    mov   eax, DWORD PTR [ebp+12]
c1 e0 08    shl   eax, 8
03 45 08    add   eax, DWORD PTR [ebp+8]
89 45 fc    mov   DWORD PTR [ebp-4], eax
8b 45 fc    mov   eax, DWORD PTR [ebp-4]

8b e5       mov   esp, ebp
5d          pop   ebp
c3          ret
```

5. (5 pts) Briefly explain what this assembly code is doing. You can use 0xA5 as an example value.

```
movsx eax, byte ptr [ebp+8]
shl   eax, 4
mov   byte ptr [ebp-1], al

movsx ecx, byte ptr [ebp+8]
sar   ecx, 4
mov   byte ptr [ebp+8], cl

movsx edx, byte ptr [ebp+8]
and   edx, 0xff
mov   byte ptr [ebp+8], dl

movsx eax, byte ptr [ebp-1]
movsx ecx, byte ptr [ebp+8]
or    eax, ecx
mov   byte ptr [ebp-1], al
```