Nombre: Alan Rojas Montt Prueba 2 Microprocesador

Problema 1:

org 100h

mov [0200h], 0069h

mov [0201h], 005Fh

mov [0202h], 0018h

mov [0203h], 0069h

mov [0204h], 0012h

mov [0205h], 00A3h

mov [0206h], 005Bh

mov [0207h], 0069h

mov [0208h], 0072h

mov [0209h], 004Ch

mov [020Ah], 001Dh

mov [020Bh], 0069h

mov [020Ch], 00B8h

mov [020Dh], 0001h

mov [020Eh], 0002h

mov [020Fh], 0003h

mov [0210h], 0004h

mov [0211h], 0005h

mov [0212h], 0006h

mov [0213h], 0007h

;---------------------

mov [0214h], 0069h

;---------------------

;--Punto de Partida---

mov bx, 0200h

mov cl, 0000h

mov dx, 0220h

step0:

mov al, [bx]

cmp al, [0214h]

jz step1

inc bx

inc cl

cmp cl, 13h

jng step0

ret

step1:

push dx

push bx

mov bx, dx

pop dx

mov [bx], dx

mov bx, dx

pop dx

add dx, 0002h

inc bx

inc cl

cmp cl, 13h

jng step0

ret

El algoritmo para resolverlo comienza con step0, en este ciclo guardamos el valor que se encuentre en la dirección apuntada por “BX” en el registro AL, luego se procede a comparar AL con el dato ubicado en [0214h] el cual es el valor a buscar.

Cada vez que su comparación active ZF = 1, da a entender que se encontró el valor a buscar dentro del rango de 00h a 13h, de esta manera salta a step1.

Ya se tenía predefinido DX como 0220h en un principio, en step1 procedemos a conmutar BX y DX, de esta manera podremos guardar en 0220h el valor de la posición de los números que se hayan repetido.

Cada vez que se encuentre o no un numero en cada posición de [BX] se incrementa CL, este registro va desde 00h a 13h, al momento de llegar a 13h el programa finalizara.

Problema 2:

org 100h

mov [0200h], 000Bh

mov [0201h], 0004h

mov [0202h], 005Ch

mov [0203h], 0009h

mov [0204h], 0031h

mov [0205h], 000Dh

mov [0206h], 00A8h

mov [0207h], 004Ch

mov [0208h], 0021h

mov [0209h], 0002h

mov bx, 0200h

mov cl, 000Ah

step0:

mov ax, 0000h

mov al, [bx]

mov dl, [bx+01]

div dl

mov [bx+10h],al

mov al, ah

mov ah, 00h

mul cl

div dl

mov [bx+11h], al

add bx, 02h

cmp bx, 020Ah

jnz step0

ret

Para resolver este problema guardamos AL el valor de [BX] y en DL el valor que se encuentre después, es decir [BX + 01], ¿Por qué?, al momento de otorgarle un valor XX a AL, tendremos que AX = 00XXh, de esta manera al utilizar DIV DL, tendremos AX / DL para el cociente y dicho dato se guardara en AL, procedemos a guardar en [BX+10h] el valor del cociente. Para agregar un decimal, debemos multiplicar el RESTO por 10 Dec ya que si dividimos asi tal cual el resto no será menor al divisor, para esto primero debemos establecer que 10 DEC = 0Ah, y AX ahora deberá ser 00YYh, donde YY representa al valor del resto obtenido en la primera división (para formar 00YYh, guardamos AH en AL y AL le damos el valor de 00h) multiplicamos AX con 0Ah y procedemos a dividir. El valor obtenido lo guardamos en [BX+11h] el cual es el espacio siguiente a [BX+10h], incrementamos BX en 02h y repetimos el proceso, al momento de que BX = 020Ah el programa finalizara.

Problema 3:

org 100h

mov [0200h], 0000h

mov [0202h], 0001h

mov [0204h], 0015h ; ejemplo con 21

mov ax, [0200h]

mov bx, [0202h]

mov bp, [0204h]

step1:

cmp bp, 01h

jz init1

jnz fibo

fibo:

call fibonacci

jmp stop

init1:

mov [0210h],0000h

ret

stop:

mov [0210h],cx

ret

fibonacci proc

add ax, bx

mov cx, ax

mov ax, bx

mov bx, cx

dec bp

cmp bp, 02h

jng stop

call fibonacci

ret

fibonacci endp

En este algoritmo creamos una subrutina Fibonacci, que suma AX con BX, siendo BX = 0001h y AX = 0000h, luego esa suma se guarda en AX, guardamos la suma en CX y a AX le damos el valor de BX, luego a BX le damos el valor de CX, y asi sucesivamente hasta el valor que nosotros deseamos calcular de la sucesión de Fibonacci. Ocupamos BP como contador y tomara el valor de [0204h], de esta forma tomara el valor que se dese buscar, cada vez que se ocupe la subrutina mencionada anteriormente ocurre que BP decrece en 01h. Sin embargo esta subrutina funciona solo para los valores desde la segunda posición en adelante, para la primera posición creamos un salto llamado step1 y init1, en step1 comparamos la posición, si esta es 01h, salta a init1 y lo guarda en [0210h] para proceder a finalizar el programa, si BP es distinto de 01h, entonces saltara a fibo: el cual contiene a la subrutina, la subrutina calculara el valor de dicha posición hasta que BP = 02h (2da posición, para la primera no cuenta esta subrutina). Al momento de llegar a BP = 02h entonces la subrutina no se vuelve a llamar a sí misma y vuelve a la línea donde se encontraba en “fibo” y procede a saltar a Stop, en stop guardara el valor calculado que se guardó en CX en la posición [0210h] y finaliza el programa.