EE 3724 Test #2 Solutions - Summer '00 – Reese

1. (33 pts) Assume the following memory contents at the start of each of the following instructions

Address	Contents															
									65 E2							
09A0:0000	C5	67	A5	00	12	BC	34	BB	E4	72	09	A3	29	01	D4	CE
09A0:0010	FE	89	02	D8	A4	8A	7C	DD	90	3C	9B	83	65	19	F6	8A
09A0:0020	A7	CC	9A	BD	8E	90	2C	59	1C	90	0E	13	8C	39	58	C6
09A0:0030	76	D7	CA	FF	D8	71	18	24	40	A8	2C	76	93	C5	0F	9E
09A0:0040	82	A6	54	2E	9A	20	0A	98	E4	A0	0E	25	38	29	2C	86

Assume the following register contents at the START of each of the following instructions.
ES: 09A0, DS: 09A0, SS: 09A1
AX = E265, DX = 73A2, CX = 0000, SP = 0018, BP=002E

Give the value of the affected register OR affected memory location after each instruction. IF MEMORY IS MODIFIED, you MUST show the modified locations on the ABOVE memory map! LIST ALL registers that are affected by the instructions except for the flag registers.

a. Push ax — sp = 0016, see map above (SS:SP = 09A1:0018 = 09A0: 0028)

b. sar dl, 2 — dl=E8h

c. shr dl,2 — dl= 28h

d. shl al,1 — al = CAh

e. xor dl, al — dl = C7h

f. or dl, F0h — dl = C7h

g. not dl — dl = F2h

h. pop dx — dx = 901C h sp=001A

i. mul dl — ax = al * dl = 101 * 162 = 16362 = 3FEA h

j. imul dl — ax = ax * dl = 101 * -94 = -9494 = DAEA h

k. and dl, 22 — dl = 22 h

2.(27 pts) Assume the same register contents/memory contents as above. For EACH of the following two instruction sequences, tell if the jump is TAKEN or NOT TAKEN.

a. cmp al,dl
   jne there TAKEN (al not equal to dl)

b. cmp al,dl
   jl there NOT_TAKEN (al, a postive number, is not less than dl, a negative number)

c. cmp al,dl
   ja there NOT_TAKEN (al is not higher than dl)

d. cmp ax,dx
   jg there NOT_TAKEN (ax, a negative number is not greater than dx, a postive number)

e. cmp ax,dx
   jb there NOT_TAKEN (ax is not below dx)

f. test al,1
   jnz there TAKEN (result of al AND 1 is non-zero, so branch taken)

g. add al,dl
   jnc there NOT_TAKEN (al+dl produces a carry, a branch not taken)

h. add al,dl
   js there NOT_TAKEN (MSB of al+dl is '0', so branch not taken)

i. add al, 40h
   jno there NOT_TAKEN (al + 40h produces an overflow, so branch not takend)

3. (3 pts) Register AL has an 8 bit value ($b_7b_6b_5b_4b_3b_2b_1b_0$). Use a single logical instruction to change the contents of AL to ($b_700000b_1b_0$). Bits B7, B1, B0 unchanged, other bits set to zero.

And al, 83h

4. (3 pts) Register AL has an 8 bit value ($b_7b_6b_5b_4b_3b_2b_1b_0$). Use a single logical instruction to change the contents of AL to ($1b_6b_5b_4b_311b_0$). Bits B7, B2,B1, set to '1', other bits unchanged.

Or al, 86h

5. (8 pts) Write a subroutine that will return the number of bytes in a string. The string is terminated by a '0' byte (count DOES NOT INCLUDE the '0' byte), and the starting address of the string is passed to the subroutine via the DS:SI register. The count should be returned as zero if the string is 'empty' (first byte is zero). The count should be returned in the AL register (maximum number of characters will be 255).

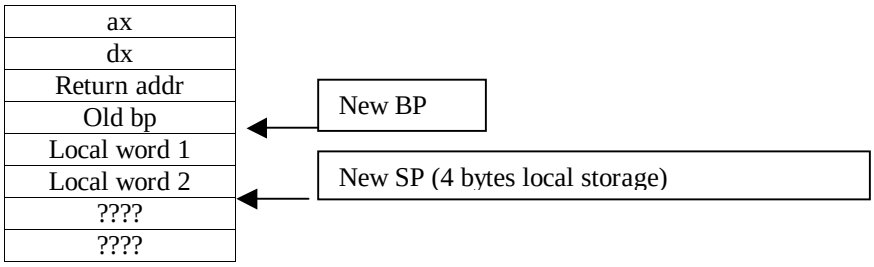
```
Strlen      proc
            Xor   ax,ax
Lp1:        mov  bl, [si]
            Cmp bl,0
            Jz          exit
            Inc al
            Inc si
            Jmp lp1
Exit:       ret
Strlen      endp
```

6. (10 pts) For the following code:

```
            push ax
            push dx
            call suba
            ….                  ;; other instructions
            ….

Suba        proc
            Enter  4,0
            …..
            …..
            leave
            ret
suba        endp
```

Draw a stack picture that shows the state of the stack after the '*enter*' instruction inside of suba. Draw the stack diagram as being 16 bits wide. Show all know registers/values that are on the stack, and show the position of BOTH the stack pointer and base pointer.



7. (8 pts) Write a subroutine that will return the maximum 16-bit SIGNED integer from an array of integers. On subroutine entry, register SI will point to the start of the array (each element is 16 bits), and register CX will have the number of integers in the array. The maximum value should be returned in the AX register. An example call to this procedure is shown below:

```
        .data
myarray  dw  -45,  1000, -34,  1500, 20,  60
count     equ   6          ; six elements in the array

        .code
        mov   ax,@data
        mov   ds,ax
        mov  cx, count
        lea    si, [myarray]
        call  findmax
        …..
        …..


findmax    proc
;;; will assume that array always has at least 1 element
        mov ax,  8000h          ;; get most negative 16-bit value into ax
lp1:    cmp ax,  [si]
        jge    skip
        mov  ax, [si]           ;; ax is less than [SI], get memory value
skip:    lea  si, [si+2]        ;; increment pointer by 2 bytes
        loop  lp1               ;; cx has count
        ret
findmax  endp
```

8. (8 pts) Write a subroutine that will take the lower 4-bits in AL (value is between 0 and 15) and convert it to its ASCII equivalent character representation (value 0 = '0' (30h), value 1 = '1' (31h), etc, value 15 = 'F' (46h). Use capital letters 'A' thru 'F' for values 10 thru 15. The ASCII character value should return in AL. You DO NOT KNOW what the upper 4-bits of register AL contains upon entry to the subroutine.

```
Mysub     proc
        And al, 0Fh       ;; mask off upper 8 bits
        Cmp  al, 09h
        Ja        skip    ;; jump if value A-F
        Add   al, 30h     ;; value is 0-9, add 30h
        Ret

Skip:   add  al, 37h      ;; value is A-F, add 37h to get to 41h – 46h
        Ret
Mysub   endp
```