EE 3724 Test #2 Solutions - Spring '01 – Reese

1. (16 pts) DS: 09A3, SS: 09A1, AX = 75D0 BX= B00C, BP:0004, SP: 248A
Give the final value of the affected register:

a. SAR BH, 1 D8h (shift arithmetic right)

b. ROR BH, 4 0Bh (rotate right)

c. SAL BH, 1 60h (shift arithmetic left)

d. What is the value of the SP after I do a 'push ax' ? SP = SP - 2 = 2488h

2. (8 pts) For each of the following mark if the branch is TAKEN or NOT TAKEN assuming the register contents in problem #1.

a. cmp ah, bh *(signed, Ah positive > BH negative, branch taken)*
jg THERE **TAKEN** NOT TAKEN

b. cmp al, bl *(unsigned, al is higher than bl, so not taken)*
jb THERE TAKEN **NOT TAKEN**

3. 12 pts. Assume a 80486 processor (32-bit data bus). Mark the value of the byte enable signals as TRUE (asserted) or FALSE (negated) for each of the following memory read operations. BE0 is for the lower 8 data lines (D7-D0), while BE3 is for the upper 8 data lines (D31-D24). Address values are in HEX!!!

	BE3	BE2	BE1	BE0
a. mov ax, [03A45] *(word operation)*	F	T	T	F
b. mov al, [03A4A] *(byte operation)*	F	T	F	F
c. mov eax, [03A40] *(double word operation)*	T	T	T	T

4. (12 pts) Assume a 80486 processor (32-bit data bus). Mark the following accesses as ALIGNED or MISALIGNED.

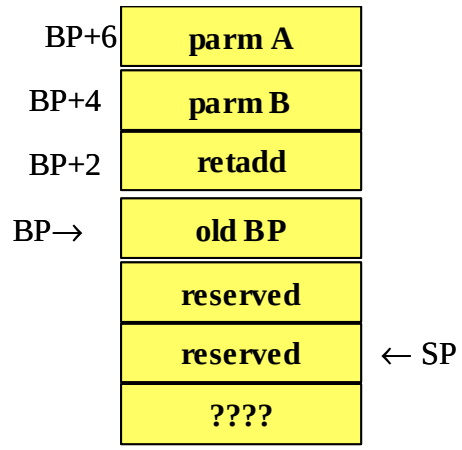
a.	mov ax, [0ADC1]	**ALIGNED**	MISALIGNED
b.	mov ax, [0ADC7]	ALIGNED	**MISALIGNED**
c.	mov eax, [0ADC6]	ALIGNED	**MISALIGNED**

5. (15 pts) Look at the code below and answer the questions

```
MAIN    push  ax        ; push PARAMETER A
        push  bx        ; push PARAMETER B
        call  SUBA
        …..

SUBA    proc
          Enter  4,0
           ;; stack frame has been setup,  POINT A
          ;;; other instructions

        ;; code needed here to clean up stack frame and passed parameters, and return to main
SUBA    endp
```



a. At 'point A', write an instruction that will read the value of parameter A into register AX. This instruction cannot change the stack (i.e, a 'POP' is incorrect answer). You can use either the SP or BP as your index register. It will help if you draw a picture of the stack.

mov ax, [bp+6]

b. Write a TWO INSTRUCTION SEQUENCE at the end of 'SUBA' that will clean up the stack frame, return to the main program, and clean up the stack of the passed parameters **A** and **B**.

```
Leave     ; clean up stack frame
ret   4   ; return and increment SP by 4
```

6. (7 pts) For a memory chip with control lines **CS**, **OE**, and **W**, what control lines must be asserted during:

a. a write operation? *CS, W*

b. a read operation? *CS, OE*

7. (15 pts) For the address decoder below:

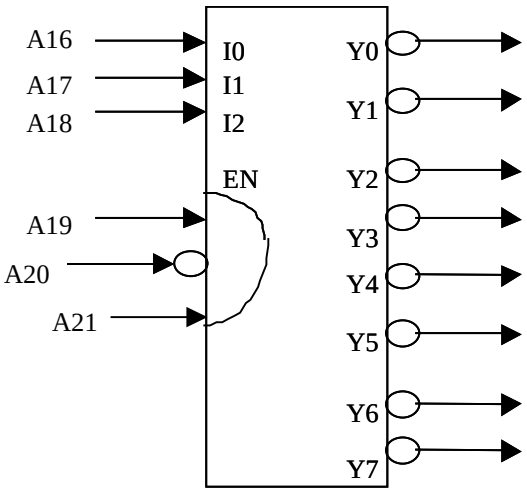
a. Give the range of addresses that output Y3 is valid for (Use HEX addresses)

b. How many TOTAL bytes is this address decoder valid for? (give the answer in Kbytes, or Mbytes)

A21	*A20*	*A19*	*A18*	*A17*	*A16*	*A15-A0*
1	*0*	*1*	*0*	*1*	*1*	*0 … 0*
1	*0*	*1*	*0*	*1*	*1*	*1 …..1*

2B0000 h to 2BFFFFh

Each decoder output is A15-A0, 2^{16}
8 decoder outputs, $2^3 * 2^{16} = 2^{19} = 2^9 * 2^{10}$ = 512K bytes



3-to-8 Decoder
I2 is MSB, I0 is LSB

8. (15 pts) Write a subroutine that will convert all lower case characters in a string to upper case. A lower case ASCII character has a value between 61H ('a') and 7Ah ('z'). To convert to lower case, subtract 20H or clear bit B5 to a zero. The starting string address is passed in register BX and the string is NULL TERMINATED (last byte is 00h). Other characters in the string that are not lower case characters should be unaffected.

```
lcase   proc

        mov   al, [bx]      ; get byte
        cmp   al, 0h        ; zero?
        Je    exit          ; exit if zero byte
        Cmp   al, 61h
        jb    skip          ; if lower than 'a', then skip
        Cmp   al, 7Ah
        Ja    skip          ; if higher than 'z', then skip
        Sub   al, 20h       ; is a lower case char, convert to upper case
        Mov   [bx], al      ; save in memory
Skip:   inc   bx            ; increment pointer to next byte
        Jmp   lcase
Exit    ret

lcase   endp
```