BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI – K K BIRLA GOA CAMPUS
II SEMESTER 2015-2016
CSF241/EEEF241/INSTRF241MICROPROCESSOR PROGRAMMING AND INTERFACING
TEST I (OPEN BOOK)

TIME: 60 Min. 24/02/2016 MM: 40

IDNO: **Name:**

Q1. In an 80486 processor that is working in 32-bit mode. For the instructions given below determine the following. **[Give Values in Hex only]** **[3]**

a) MOV AX, ES:[DI+BP+04_H]

Addressing Mode	Base relative plus Indexed
Machine Code	6766268B4304

Q2. Replace the following program segments by **a single instruction** of 80486.

[Clarification: Each program segment achieves a certain final result. You need to give a single instruction that will achieve the same result. The single instruction needs only achieve the final result] **[3+4]**

	Program	**Instruction**		**Program**	**Instruction**
A	**mov cx,[bx]** **bt cx,15** **jnc x1** **or ecx,0ffff0000h** **jmp x2** **x1: and ecx,0000ffffh** **x2:**	**movsx ecx,[bx]**	**B**	**push ax** **pushf** **pop ax** **xor ax,0001h** **push ax** **popf** **pop ax**	**cmc**

Q3. For the following Instructions what will be the machine cycles executed by 8086. Enter the machine cycles that will be executed in proper order. **[8]**

	Instruction	Cycles		Instruction	Cycles
A	MOVSW	1 MEMR – opcode 1 MEMR – data 1 MEMW - data	B	STD	1 MEMR - opcode
C	MOV WORD PTR[2000_H], 1000_H	1 MEMR – opcode 1 MEMR – displacement 1 MEMR – imm data 1 MEMW data	D	PUSH [$011C_H$]	1 MEMR -opcode 1 MEMR -displacement 1 MEMR - data 1 MEMW - stack

Marks on this page

Q4. If an 8086 processor is working at 10 MHz and the memory access time is 400ns. The number of wait states required will be ______3______ , considering an address set-up time of 110ns, data set-up time of 40ns with a latching and buffer delays of 30ns. **[2]**

Q5. In an 80486 processor that is working in real mode. Suppose that

EAX	**11112222$_H$**	**ECX**	**55556666$_H$**
EDX	**77778888$_H$**	**EBX**	**33334444$_H$**
ESP	**00003000$_H$**	**EBP**	**00009999$_H$**
ESI	**0000AAAA$_H$**	**EDI**	**0000BBBB$_H$**

What will be the effect of executing the following code snippet on an 80486 processor? Fill in the table given below **[8]**

```
PUSH EBP
PUSHAD
POPA
POP ECX
POP CX
```

EAX	**1111 0000**	**ECX**	**3333 8888**
EDX	**7777 0000**	**EBX**	**3333 9999**
ESP	**0000 2FF2**	**EBP**	**0000 AAAA**
ESI	**0000 0000**	**EDI**	**0000 BBBB**

Q6. Write an 80486 ALP that will examine a series of memory locations storing ***16-bit signed numbers***. The numbers are stored starting from location ***loc1***. The program should separate the positive numbers and store them in memory starting from location ***pos1*** and the negative numbers in memory location starting from ***neg1.*** The count of memory locations to be examined is stored in ***cnt1*** and will not exceed 250$_d$. The checking and the separation must be done by a sub-routine named ***sep1***. The number to be examined should be passed to sub-routine using SI as pointer. The main program only does the initialization and looping. **[12]**

Eg. If

loc1 : ***-200,200,300,500,-700,900,-100***

after program

neg1: ***-200 ,-700,-100***

pos1: ***200,300,500,900***

[YOU CAN USE THE BLANK SPACE AT THE BACK OF THIS PAGE FOR WRITING THE PROGRAM]

Marks on this page

```
.model  tiny
.486
.data
loc1    dw          -1,2,3,-7,5,-90,78
cnt1    db          7
pos1    dw          7 dup(?)
neg1    dw          7 dup(?)
st1     dw          10 dup (?)
st2     dw          ?
.code
.startup
        Lea         sp,st2
        lea         si,loc1
        lea         di,pos1
        lea         bx,neg1
        movzx       cx,cnt1
x1:     call        sep1
        loop        x1
.exit
sep1    proc        near
        lodsw
        bt          ax,15
        jc          x2
        stosw
        jmp         x3
x2:     mov         [bx],ax
        inc         bx
        inc         bx
x3:     ret
sep1    endp
end
```

Correction Rubric for Program

Program structure – model definition, .486 definition, .code, .startup, .exit, end – all in proper place -1M
Definition of sub program – proc near, endp – name of subprogram should be sep1 -1M
Data declarations – loc1 with dw, cnt1 with db and reserving word locations for neg1 and pos1 – 1 M
Initialize stack and SP – 1 M
Initializing pointers (if 16-bit addressing used only si,di,bx,bp allowed), load count as 8-bit or 16-bit using zero extension – 1M
Calling subroutine with SI pointer -1M
Checking if no is positive or negative – has to be word op with word data transfer -2M
Storing in positive location – 1M
Storing in negative location – 1M
Update Source and Destination Pointer for word operation – 1M
Return and proper looping - 1 M

Marks on this page