MIDTERM
CS401 Assembly Language

Q: Affected flag of AND operation marks 2
Answer:-
Affected Flag of AND are:
CF, OF, PF, SF, ZF and AC

Q: Relation between RET and CALL is dependent or independent. Marks 2
Answer:- (Page 64)
Technically they are independent in their operation but both the instructions are commonly used as a pair. The RET works regardless of the CALL and the CALL works regardless of the RET.

Q: What is the difference between LES and LDS instruction? marks 3
Answer:- (Page 97)
LES loads ES while LDS loads DS. These instructions have two parameters, one is the general purpose register to be loaded and the other is the memory location from which to load these registers.

MIDTERM
CS401 Assembly Language

21. Write down the procedure to clear the selective bit. (Marks: 2)
Answer:- (Page 59)
The source operand is loaded with a mask containing one at positions which are retain their old value and zero at positions which are to be zeroed. The effect of applying this operation on the destination with mask in the source is to clear the desired bits. This operation is called masking.

22. Why REP prefix is generally not used with LODS instruction? (Marks: 2)
Answer:- (Page 92)
LODS is generally used in a loop and not with the REP prefix since the value previously loaded in the register is overwritten if the instruction is repeated and only the last value of the block remains in the register.

23. What is difference between REPE and REPNE? (Marks: 3)
Answer:- (Page 93)
REPE or REPZ repeat the following string instruction while the zero flag is set and REPNE or REPNZ repeat the following instruction while the zero flag is not set.

24. Describe Push and Pop with the help of an example. (Marks: 3)
Answer:- (Page 68)
The operations of placing items on the stack and removing them from there are called push and pop. Memory is like a shelf numbered as zero at the top and the maximum at the bottom. If a decrementing stack starts at shelf 5, the first item is placed in shelf 5; the next item is placed in shelf 4, the next in shelf 3 and so on.

25. Explain all characteristics of SCAS instruction. (Marks: 5)
Answer:- (Page 92)
SCAS compares a source byte or word in register AL or AX with the destination string element addressed by ES:DI and updates the flags. DI is updated to point to the next location. SCAS is often used to locate equality or in-equality in a string through the use of an appropriate prefix.
SCAS is a bit different from the other instructions. This is more like the CMP instruction in that it does subtraction of its operands. The prefixes REPE (repeat while equal) and REPNE (repeat while not equal) are used with this instruction. The instruction is used to locate a byte in AL in the block of memory. When the first equality or inequality is encountered; both have uses.

26. Describe Local Variable? (Marks: 5)
Answer:- (Page 75)
Another important role of the stack is in the creation of local variables that are only needed while the subroutine is in execution and not afterwards. They should not take permanent space like global variables. Local variables should be created when the subroutine is called and discarded afterwards. So that the spaced used by them can be reused for the local variables of another subroutine. They only have meaning inside the subroutine and no meaning outside it.
The most convenient place to store these variables is the stack. We need some special manipulation of the stack for this task. We need to produce a gap in the stack for our variables.

MIDTERM
CS401 Assembly Language

1. Which flag affected by often and operator
Answer:- rep

2. Explain divide overflow error.
Answer:- (Page 85)
If a large number is divided by a very small number it is possible that the quotient is larger than the space provided for it in the implied destination. In this case an interrupt is automatically generated and the program is usually terminated as a result. This is called a divide overflow error.

3. For what purpose "INT4" is reserved.
Answer:- (Page 106)
INT4 is reserved for Arithmetic Overflow, change of sign bit

4. Double working of Exchange instruction with help of an example.
Answer:- (Page 73)
XCHG (Exchanges) instruction swaps the contents of the destination (first) and source (second) operands. The operands can be two general-purpose registers or a register and a memory location. If a memory operand is referenced, the processor's locking protocol is automatically implemented for the duration of the exchange operation, regardless of the presence or absence of the LOCK prefix or of the value of the IOPL.
xchg ax, [bx+si+2] ; exchange ax with second number

5. What is the difference between LES and LDS instruction?
Answer:- rep

MIDTERM
CS401 Assembly Language

1. Two form of Moves introduction
Answer: (Page 98)
MOVS has the two forms MOVSB and MOVSW

2. Describe push & pop with help of example
Answer:- (Page 71)
PUSH
PUSH decrements SP (the stack pointer) by two and then transfers a word from the source operand to the top of stack now pointed to by SP. PUSH often is used to place parameters on the stack before calling a procedure; more generally, it is the basic means of storing temporary data on the stack. Consider the example of test tube.(see stack)

POP
POP transfers the word at the current top of stack (pointed to by SP) to the destination operand and then increments SP by two to point to the new top of stack. POP can be used to move temporary variables from the stack to registers or memory. Observe that the operand of PUSH is called a source operand since the data is moving to the stack from the operand, while the operand of POP is called destination since data is moving from the stack to the operand.

3. How value of Ip Register is saved & updated when call. Instruction executed.
Answer:- (Page 71)
When Call instruction is executed the stack pointer (SP) is decremented by 2 and value of IP is pushed onto stack.

4. AX = 0x3412, BX=0x7856, CX= 0x1CAB, Sp=0x100 Give contents of AX, BX,CX,SP After creating instruction push AX, Push BX, cHG, AX,CX, POP CX

5. Explain Divide over Flows Error.
Answer:- rep

6. How many parameters a subroutine can receive through stack.
Answer:- (Page 72)
The maximum parameters a subroutine can receive are seven

MIDTERM
CS401 Assembly Language

what is push and pop explain with example 3marks
Answer:- rep

how the string instruction is used in the block of memory 3
Answer: (Page 92)
STOS transfers a byte or word from register AL or AX to the string element addressed by ES:DI and updates DI to point to the next location. STOS is often used to clear a block of memory or fill it with a constant.

affected flag of AND operation 2marks
Answer:- rep

how to convert the 3 dimensional memory in 1 dimensional memory

write the scas instruction 5marks
Answer:- rep

Midterm Paper

1. How many block processing instructions in 8088, just name the number of processing instructions?
Answer:- (Page 91)
There are just 5 block processing instructions in 8088. The five instructions are STOS, LODS, CMPS, SCAS, and MOVS called store string, load string, compare string, scan string, and move string respectively

2. What are the instruction use by assembly language for permanent diversion and temporarily diversion?
Answer:- (Page 64)
The instructions for permanent diversion in 8088 are the jump instructions, while the instruction for temporary diversion is the CALL instruction.

3. How much string instructions are in assembly language? Write down all instructions.
Answer:- (Page 83)
STOS – Clearing the Screen
LODS – String Printing
SCAS -String Length
LES and LDS
MOVS -Screen Scrolling
CMPS – String Comparison

4. What is the purpose of following string instructions?
a. STOS
b. CMPS
Answer:- (Page 92)
STOS:-
STOS transfers a byte or word from register AL or AX to the string element addressed by ES:DI and updates DI to point to the next location. STOS is often used to clear a block of memory or fill it with a constant.

CMPS:-
CMPS subtracts the source location DS:SI from the destination location ES:DI. Source and Destination are unaffected. SI and DI are updated accordingly. CMPS compares two blocks of memory for equality or inequality of the block. It subtracts byte by byte or word by word.

Midterm Paper

What is the syntax of PUSH instruction? 2 Marks
Answer:- (Page)
push ax

For what purpose DS and ES registers are generally used in the context of video memory. 2 Marks
Answer:- (Page 81)
Both DS and ES can be used to access the video memory. However we commonly keep DS for accessing our data, and load ES with the segment of video memory.

When the instructions "push ax" is executed in decrementing stack how the value of SP will change. 3 Marks
Answer: (Page 71)
When the instructions "push ax" is executed in decrementing stack, SP is decremented by two and IP is pushed onto the stack.

What colors are repeated by 0th, 1st and 2nd bits (from least significant side) in Attribute Byte? 3 Marks
Answer:- (Page 81)
blue , green , Red

How SCAS instructions can be used to detect Null at the end of a string? 5 Marks
Answer:-Page 95 (Not Sure)
We use SCASB with REPNE and a zero in AL to find a zero byte in the string. In CX we load the maximum possible size, which are 64K bytes. However actual strings will be much smaller. An important thing regarding SCAS and CMPS is that if they stop due to equality or inequality, the index registers have already incremented. Therefore when SCAS will stop DI would be pointing past the null character.

Midterm Paper

Q.No1:
What are instructions use by the Assembly language for permanent and temporary division? (2 marks)
Answer:- rep

Q.No2: From where does the contents of SL and DS registers are loaded as a result of execution of the instruction of the instruction" LDS SL [BP +4]"? (2 marks)
Answer: (Page 97)
"lds si, [bp+4]" will load SI from BP+4 and DS from BP+6.

Q.No:3 .How 16-bit MUL operations is different from 32-bit MUL operation? (5 marks)
Answer: (Page 65)
If the operands were 16bit the answer would be in 32bit anf if the if the operands were 32bit the answer would be 64bit.
For declaration of 16 bit we use "dw" and for 32-bit we use "dd".

Q.No:4.Describe the working of the CALL instruction with the reference of Stack. (3 marks)
Answer:- (Page 68)
During the CALL operation, the current value of the instruction pointer is automatically saved on the stack, and the destination of CALL is loaded in the instruction pointer. Execution therefore resumes from the destination of CALL.

Q.No5: How many BYTES will be move by each of the following block of codes? (3 marks)
a) MOV cx,384
REP movsb

b) MOV cx,384
REP movsw

Q.No:6 Consider the following pseudo code and write the corresponding assembly code for it
Note: There is more credit for a shorter code (05 marks)
If (al > C1) AND (b1 > a1)
{
dx=1
}

Midterm Paper

21.Explain the function of rotate right (ROR) instruction

Answer:- (Page 53)

In the rotate right operation every bit moves one position to the right and the bit dropped from the right is inserted at the left. This bit is also copied into the carry flag.

22. Why REP prefix is generally not used with LODS instruction?

Answer:- rep

23. Write all steps of algorithm for printing number 352.

Answer:- (Page 84)

The steps of our algorithm are outlined below.

- Divide the number by base (10 in case of decimal)
- The remainder is its right most digit
- Convert the digit to its ASCII representation (Add 0x30 to the remainder in case of decimal)
- Save this digit on stack
- If the quotient is non-zero repeat the whole process to get the next digit, otherwise stop
- Pop digits one by one and print on screen left to right

24. What are the result after performing the instruction (each carry 1 marks)

1. and ax,bx
2. or ax,bx
3. xor ax,bx

Given that ax = 00110011 and bx = 00010001

Answer:-

0x0011
0x0033
0x0022

25. Describe Local Variables?

Answer:- rep

26. Explain the complete operation of Interrupt when it is generated.

Answer:- (Page 104)

Interrupt is the result of an INT instruction (software interrupt) or it is generated by an external hardware which passes the interrupt number by a different mechanism. The currently executing instruction is completed, the current value of FLAGS is pushed on the stack, then the current code segment is pushed, then the offset of the next instruction is pushed. After this it automatically clears the trap flag and the interrupt flag to disallow further interrupts until the current routine finishes. After this it loads the word at nx4 in IP and the word at nx4+2 in CS if interrupt n was generated. As soon as these values are loaded in CS and IP execution goes to the

start of the interrupt handler. When the handler finishes its work it uses the IRET instruction to return to the caller. IRET pops IP, then CS, and then FLAGS. The original value of IF and TF is restored which re-enables further interrupts

Midterm Paper

Q#21 Mark 2
What are the instructions used by assembly language for permanent and temporary diversions.
Answer:- rep

Q#22 Which instruction is used to determine zero bit in string. Mark 2
Answer: (Page 94)
REPE or REPZ repeat the following string instruction while the zero flag is set and REPNE or REPNZ repeat the following instruction while the zero flag is not set.

Q#23 Explain the use of TEST instruction. Mark 3
Answer:- (Page 104)
The test instruction is used for bit testing. BX holds the mask and in every next iteration it is shifting left, as our concerned bit is now the next bit.

Q#24 Explain LES and LDS Mark 3
Answer:- rep

Q#25 Describe local variables. Mark 5
Answer:- rep

Q#26 Describe MOVS and CMPS instructions Mark 5
Answer:- (Page 92)
MOVS
MOVS transfers a byte or word from the source location DS:SI to the destination ES:DI and updates SI and DI to point to the next locations. MOVS is used to move a block of memory. The DF is important in the case of overlapping blocks.

CMPS
CMPS subtracts the source location DS:SI from the destination location ES:DI. Source and Destination are unaffected. SI and DI are updated accordingly. CMPS compares two blocks of memory for equality or inequality of the block. It subtracts byte by byte or word by word. If used with a REPE or a REPNE prefix is repeats as long as the blocks are same or as long as they are different.

MIDTERM EXAMINATION

Question No: 17 (Marks: 2)
Define short jump
Answer:- (Page 46)
If the offset is stored in a single byte as in 75F2 with the opcode 75 and operand F2, the jump is called a short jump.

Question No: 18 (Marks: 2)
Every character is displayed on the screen in the form of a word. what each byte of this word represents?
Answer: (Page 81)
The second byte in the word designated for one screen location holds the foreground and background colors for the character.
So the pair of the ASCII code in one byte and the attribute in the second byte makes the word that corresponds to one location on the screen. The lower address contains the code while the higher one contains the attribute.

Question No: 19 (Marks: 2)
IF DF=0 what its represent and IF DF=1 what its represent?

Answer: Page 17
This flag tells whether the current operation has to be done from bottom to top of the block (D=0) or from top to bottom of the block (D=1).

Question No: 20 (Marks: 3)

When the instruction "push ax" is executed in decrementing stack how the value of SP will change

Answer:- rep

Question No: 21 (Marks: 3)
Explain LES and LDS instructions.
Answer:- rep

Question No: 22 (Marks: 5)
Explain how extended shifting is performed
Answer:- (Page 56)
Using our basic shifting and rotation instructions we can effectively shift a 32bit number in memory word by word. We cannot shift the whole number at once since our architecture is limited to word operations. The algorithm we use consists of just two instructions and we name it extended shifting.
num1: dd 40000
shl word [num1], 1
rcl word [num1+2], 1
The DD directive reserves a 32bit space in memory, however the value we placed there will fit in 16bits. So we can safely shift the number left 16 times. The least significant word is accessible at num1 and the most

significant word is accessible at num1+2.
The two instructions are carefully crafted such that the first one shifts the lower word towards the left and the most significant bit of that word is dropped in carry. With the next instruction we push that dropped bit into the least significant bit of the next word effectively joining the two 16bit words. The final carry after the second instruction will be the most significant bit of the higher word, which for this number will always be zero.

Question No: 23 (Marks: 5)
Explain MUL instruction in both cases (i) if the source operand is byte (ii) if the source operand is a word?
Answer:- (Page 87)
MUL (multiply) performs an unsigned multiplication of the source operand and the accumulator. If the source operand is a byte, then it is multiplied by register AL and the double-length result is returned in AH and AL. If the source operand is a word, then it is multiplied by register AX, and the double-length result is returned in registers DX and AX.

MIDTERM EXAMINATION

Question No: 17 (Marks: 2)
Why is it necessary to provide the segment and offset address in case of FAR jump ?
Answer:- (Page 46)
Far jump must be used a two byte segment and a two byte offset are given to it Because It loads CS with the segment part and IP with the offset part.

Question No: 18 (Marks: 2)
What's your understanding about Incrementing and Decrementing Stack?
Answer:- (Page 68)
A decrementing stack moves from higher addresses to lower addresses as elements are added in it while an incrementing stack moves from lower addresses to higher addresses as elements are added.

Question No: 19 (Marks: 2)
IF DF=0 what its represent and IF DF=1 what its represent ?

Answer:- rep

Question No: 20 (Marks: 3)
What is the Difference between CALL and RET
Answer:- (Page 64)
The CALL instruction allows temporary diversion and therefore reusability of code. The word return holds in its meaning that we are to return from where we came and need no explicit destination.
Therefore RET takes no arguments and transfers control back to the instruction following the CALL that took us in this subroutine.

Question No: 21 (Marks: 3)
Tell the Formula to scroll up the screen
rep movsw **scroll up**

Answer:- (Page 99)

```
scrollup: push bp
mov bp,sp
push ax
push cx
push si
push di
push es
push ds
mov ax, 80 ; load chars per row in ax
mul byte [bp+4]                 ; calculate source position
mov si, ax                      ; load source position in si
push si                         ; save position for later use
shl si, 1                       ; convert to byte offset
mov cx, 2000                    ; number of screen locations
sub cx, ax                      ; count of words to move
mov ax, 0xb800
mov es, ax                      ; point es to video base
mov ds, ax                      ; point ds to video base
xor di, di                      ; point di to top left column
cld                             ; set auto increment mode
rep movsw                       ; scroll up
mov ax, 0x0720                  ; space in normal attribute
pop cx                          ; count of positions to clear
rep stosw                       ; clear the scrolled space
pop ds
pop es
pop di
pop si
pop cx
pop ax
pop bp
ret 2
```

Question No: 22 (Marks: 5)
Explain how extended shifting is performed
Answer:- (Page 56)
Using our basic shifting and rotation instructions we can effectively shift a 32bit number in memory word by word. We cannot shift the whole number at once since our architecture is limited to word operations. The algorithm we use consists of just two instructions and we name it extended shifting.

```
num1: dd 40000
shl word [num1], 1
rcl word [num1+2], 1
```

The DD directive reserves a 32bit space in memory; however the value we placed there will fit in 16bits. So we can safely shift the number left 16 times.
The least significant word is accessible at num1 and the most significant word is accessible at num1+2.
The two instructions are carefully crafted such that the first one shifts the lower word towards the left and the most significant bit of that word is dropped in carry. With the next instruction we push that dropped bit into the least significant bit of the next word effectively joining the two 16bit words.
The final carry after the second instruction will be the most significant bit of the higher word, which for this number will always be zero.

Question No: 23 (Marks: 5)
Write a subroutine to calculate the string length?
Answer:- (Page 97)

```
subroutine to calculate the length of a string
; takes the segment and offset of a string as parameters
strlen: push bp
mov bp,sp
push es
push cx
push di
les di, [bp+4]          ; point es:di to string
mov cx, 0xffff          ; load maximum number in cx
xor al, al              ; load a zero in al
repne scasb             ; find zero in the string
mov ax, 0xffff          ; load maximum number in ax
sub ax, cx              ; find change in cx
dec ax                  ; exclude null from length
pop di
pop cx
pop es
pop bp
ret 4
```

MIDTERM EXAMINATION

Question No: 17 (Marks: 2)
What is difference between SHR and SAR instructions?
Answer:- (Page 52)
The shift logical right operation inserts a zero from the left and moves every bit one position to the right and copies the rightmost bit in the carry flag.

Question No: 18 (Marks: 2)
For what purpose "**INT 1**" is reserved?
Answer:- (Page 105)
INT 1, Trap, Single step Interrupt This interrupt is used in debugging with the trap flag. If the trap flag is set the Single Step Interrupt is generated after every instruction. By hooking this interrupt a debugger can get control after every instruction and display the registers etc. 8088 was the first processor that has this ability to support debugging.

Question No: 19 (Marks: 2)
Define implied operand?
Answer:- (Page 18)
An implied operand means that it is always in a particular register say the accumulator, and it need not be mentioned in the instruction.

Question No: 20 (Marks: 3)
Describe the working of the CALL instruction with the reference of Stack.
Answer:- rep

Question No: 21 (Marks: 3)
Tell the Formula to scroll up the screen
rep movsw scroll up
Answer:- rep

Question No: 22 (Marks: 5)
What is the difference between LES and LDS instructions?
Answer:- rep

Question No: 23 (Marks: 5)
Explain the process of ADC?
Answer:- (Page 57)
The ADC instruction is specifically placed for extending the capability of ADD. Numbers of any size can be added using a proper combination of ADD and ADC. All basic building blocks are provided for the assembly language programmer, and the programmer can extend its capabilities as much as needed by using these fine instructions in appropriate combinations. Further clarifying the operation of ADC, consider an instruction "ADC AX, BX." Normal addition would have just added BX to AX, however ADC first adds the carry flag to AX and then adds BX to AX. Therefore the last carry is also included in the result.

IF DF=0 what its represent sssand IF DF=1 what its represent ?

Answer:- rep

Relation between RET and CALL is dependent or independent.
Answer:- rep

Write all steps of algorithm for printing number 352.
Answer:- rep

When the instruction "push ax" is executed in decrementing stack how the value of SP will change
Answer:- rep

Explain MUL instruction in both cases (i) if the source operand is byte (ii) if the source operand is a word?
Answer:- rep

How string instructions work on the block of memory?
Answer:- rep

What is mismatch? Define with one example.
Answer: Page 29
If we write our code in bytes but we declare in word then there will be a logical error in the code and It is called size mismatch error

```
; a program to add three numbers using byte variables
[org 0x0100]
mov al, [num1] ; load first number in al
mov bl, [num1+1] ; load second number in bl
add al, bl ; accumulate sum in al
mov bl, [num1+2] ; load third number in bl
add al, bl ; accumulate sum in al
mov [num1+3], al ; store sum at num1+3
mov ax, 0x4c00 ; terminate program
int 0x21
num1: dw 5, 10, 15, 0
```

Define the LES and LDS instructions do?
Answer:- rep

Write a subroutine to clear screen.
Answer:- (Page 93)

```
; clear screen using string instructions
[org 0x0100]
        jmp start

; subroutine to clear the screen
clrscr: push es
push ax
push cx
push di
mov ax, 0xb800
mov es, ax        ; point es to video base
```

```
xor di, di        ; point di to top left column
mov ax, 0x0720    ; space char in normal attribute
mov cx, 2000      ; number of screen locations
cld               ; auto increment mode
rep stosw         ; clear the whole screen
pop di
pop cx
pop ax
pop es
ret
start: call clrscr   ; call clrscr subroutine
mov ax, 0x4c00       ; terminate program
int 0x21
```

Replace the following invalid instruction with the single valid instruction

(a)mov IP

(b)mov IP,L5

a.mov IP

Answer:-

RET:-The instruction Ret will pop the Ip address.

b.mov IP,L5

Answer:-

CALL:-The instruction Ret will pop the Ip address.