Darryl Fernandes 2020012004 (72)

Assignment 02

Q1)

→ Each string instruction may require a source operand, a ~~f~~ destination operand or both. For 32 bit segment, String Instructions use ESI & EDI registers to point to the source & destinations operands respectively. For 16-bit segments, however the SI & the DI registers are used to point to the source & destinations, respectively. There are five basic Instruction for processing strings. They are as fallows:

1) MOVS: This instruction moves 1 byte. Word or Double word of data from memory location to another

eg:

```
MOV AX, 0000H
MOV DS, AX  : Initialize dataseg
            : register to 0
```

2) LODS: This instruction loads from memory. If the operand is of one byte. It is loaded into AL register, if the operand is one word. It is loaded into the AX register & a double word is loaded into the EAX register.

eg:

```
MOV SI, OFFSET S_STRING;
  point SI at string
LODS S_STRING
```

3) STOS - This instructions stores data from regester (AL , AX or EAX) to memory

eg:

```
MOV DI ; OFFSET D_STRING
STOS D_STRING
```

4) CMPS : This instructions compares two data items in memory. Data could be of a byte size, word, or doubleword.

eg:

```
MOV SI , OFFSET F_STRING;
MOV DI , OFFSET S_STRING;
CLD
CMPS F_STRING , S_STRING;
```

5) SCAS : This instructions compares the context of a registor (AL, AX or EAX) with the context of an item in memory.

eg:

```
MOV AL , 0DH
MOV DL , OFFSET TEXT_STRING
MOV CX , 80
CLD
REPNE SCAS TEXT_STRING
```

Q2) REP :

Repeat String Operation (rep, repnz, repz)

Operation : Repeat String operation until tested condition · Use the rep (repeat while equal), rpnz (repeat while nonzero) or rpz (repeat while zero) prefixes causes the associated string instruction to repeat until the count register (CX) or the zero flag (ZF) matches a tested condition

eg: Repeat while equal: Copy the 8bit byte from the DS:[(E)SI] to the ES [(E)DI] register re pi

JC :

The JC instruction branches to the specific address if the carry flag is set · otherwise executing continues with the next instruction No flags are affeted by thes instructions

eg:- JC LABEL

JMP:

The JMP instruction transfers execution to the address generated by adding the 8 bit value in the accumulator to the 16-bit value in the DPTR register are

altered. No flags are affeted by this instructio
eg: JMP QA + DPTR

CALL: The call instructions is used whenera we need to make a call to some procedure or a sub-program. Whenever a call is made, the process takes place inside the microprocessor. The address of the next instruction that exits in the caller problem

eg:- CALL subprogramme_name

WAIT: When this instruction is executed, the 8086 enters an ~~ind~~ idle conditions in which it is doing processing wait will stay in this idle state until test input pin is made low or until an interuppt signal is recevied or INTR or the NMI intercept input pins. The WAIT instructions is used to synchronize with external hardware such as the math coprocessor. WAIT does not affect any flag
eg: WAIT