SCAS: Scan String BYTE or String Word: This instruction scans a string of bytes or words for an operand byte or word specified in the register AL or AX. The string is pointed to by ES:DI register pair. The length of the string is stored in CX. The DF controls the mode for scanning of the string as stated in case of MOVSB instruction. Whenever a match to the specified operand, is found in the string, execution stops and the zero flag is set. If no match is found, the zero flag is reset. The REPNE prefix is used with the SCAS instruction. The pointers and counters are updated automatically, till a match is found.

Ex:

```
MOV AX, SEG        ; Segment address of the string, i.e. SEG is moved to AX.
MOV ES, AX         ; Load it to ES.
MOV DI, OFFSET     ; String offset, i.e. OFFSET is moved to DI.
MOV CX,010H        ; Length of the string is moved to CX.
MOV AX, WORD       ; The word to be scanned for, i.e. WORD is in AL.
CLD                ; Clear DF
REPNE SCASW        ; Scan the 010H bytes of the string, till a match to WORD is found.
```

This string of instructions finds out, if it contains WORD. IF the WORD is found in the word string, before CX becomes zero, the ZF is set, otherwise the ZF is reset. The scanning will continue till a match is found. Once a match is found the execution of the program proceeds further.

LODS: Load string Byte or String word: The LODS instruction loads the AL/AX register by the content of a string pointed to by DS:SI register pair. The SI is modified automatically depending on DF. If it is a byte transfer (LODSB), the SI is modified bye one and if it is a word transfer (LODSW), the SI is modified by two. No other flags are affected by this instruction.

STOS: Store String Byte or String Word: The STOS instruction stores the AL/AX register contents to a location in the string pointed by ES:DI register pair. The DI is modified Accordingly. No flags are affected by this instruction.

The direction flag controls the string instruction execution. The source index SI and destination index DI are modified after each iteration automatically. If DF=1, then the execution follows auto decrement mode. In this mode, SI and DI are decremented automatically after each iteration (by1 or 2 depending on byte or word operations) Hence, in auto decrementing mode, the string are referred to by their ending addresses. If DF=0, then the execution follows auto increment mode. In this mode, SI and DI are incremented automatically (by 1 or 2 depending on byte or word operation) After each iteration, hence the strings, in this case, are referred to by their starting addresses.

7.Control Transfer or Branching Instruction:

The control transfer instructions transfer the flow of execution of the program to a new address specified in the instruction directly or indirectly. When this type of instruction is executed, the CS and IP registers get loaded with new values of CS and IP corresponding to the location where the flow of execution is going to be transferred.

This type of instructions are classified in two types:

i. **Unconditional control Transfer (Branch) Instructions:**
In case of unconditional control transfer instructions; the execution control is transferred to the specified location independent of any status or condition. The CS and IP are unconditionally modified to the new CS and IP.

ii. **Conditional Control Transfer (Branch) Instructions:**
In the conditional control transfer instructions, the control is transferred to the specified location provided the result of the previous operation satisfies a particular condition, otherwise, the execution continues in normal flow sequence. The results of the previous operations are replicated by condition code flags. In other words, using type of instruction the control will be transferred to a particular specified location, if a particular flag satisfies the condition.

Unconditional Branch Instructions:

CALL	**RET**	**JUMP**	**IRET**
INT N	**INT O**	**LOOP**	

Scanned with CamScanner