Important Assembler Directives of the 8086 Microprocessor

- Data declaration directives: DB, DW, DD, DQ, DT
- ASSUME
- END directives
- EQU Directive
- PROC
- ORG
- SEGMENT
- GROUP, INCLUDE, EVEN, ALIGN
- EXTRN, PUBLIC,
- TYPE, PTR,
- LENGTH, OFFSET
- NAME, LABEL, SHORT, GLOBAL

Data declaration directives

1. DB - The DB directive is used to declare a BYTE -2-BYTE variable - A BYTE is made up of 8 bits.

- examples:
  - PRICES DB 49H, 98H, 29H; Declare array of 3 bytes named PRICES and initialize them with specified values
  - TEMP DB 100 DUP (?); Set aside 100 bytes of storage in memory and give it the name TEMP. But leave the 100 bytes un-initialized

2. DW - The DW directive is used to declare a WORD type variable - A WORD occupies 16 bits or (2 BYTE).

- examples:
  - WORDS DW 1234H, 3456H; Declares an array of 2 words and initialize them with the specified values

3. DD - The DD directive is used to declare a DWORD - A DWORD double word is made up of 32 bits =2 Word's or 4 BYTE.

- examples:
  - ARRAY DD 25629261H; This will define a double word named ARRAY and initialize the double word with the specified value when the program is loaded into memory to be run. The low word, 9261H, will be put in memory at a lower address than the high word.

Data declaration directives (cont.)

- **DQ (DEFINE QUADWORD)**
  The DQ directive is used to tell the assembler to declare a variable 4 words in length or to reserve 4 words of storage in memory.
  - **Example:**
    BIG_NUMBER DQ 243598740192A92BH; This will declare a variable named BIG_NUMBER and initialize the 4 words set aside with the specified number when the program is loaded into memory to be run.

- **DT (DEFINE TEN BYTES)**
  The DT directive is used to tell the assembler to declare a variable, which is 10 bytes in length or to reserve 10 bytes of storage in memory.
  - **Example:**
    PACKED_BCD DT 11223344556677889900

    This will declare an array named PACKED_BCD, which is 10 bytes in length. It will initialize the 10 bytes with the values 11, 22, 33, 44, 55, 66, 77, 88, 99, and 00 when the program is loaded into memory to be run. The statement RESULT DT 20H DUP (0) will declare an array of 20H blocks of 10 bytes each and initialize all 320 bytes to 00 when the program is loaded into memory to be run.

ASSUME DIRECTIVE

- ASSUME Directive - The ASSUME directive is used to tell the assembler that the name of the logical segment should be used for a specified segment. The 8086 works directly with only 4 physical segments: a Code segment, a data segment, a stack segment, and an extra segment.

Example:

- ASSUME CS:CODE ;This tells the assembler that the logical segment named CODE contains the instruction statements for the program and should be treated as a code segment.

- ASSUME DS:DATA ;This tells the assembler that for any instruction which refers to a data in the data segment, data will found in the logical segment DATA.

End directive

- END - End Program
- ENDP - End Procedure
- ENDS - End Segment

- END – it signifies the end of the program module
  - The assembler will ignore any statement after an END directive
- ENDP - indicates the end of a procedure
  - Syntax: Procedure_name ENDP
- ENDS - indicates the end of a logical segment
  - Syntax: Segment_name ENDS

Equate (EQU) Directive

- EQU - This EQU directive is used to give a name to some value or to a symbol. Each time the assembler finds the name in the program, it will replace the name with the value or symbol you given to that name.
- Example:
  - FACTOR EQU 03H ; you has to write this statement at the starting of your program
  - later in the program you can use this as follows :
  - ADD AL, FACTOR ; When it codes this instruction the assembler will code it as ADDAL, 03H ;The advantage of using EQU in this manner is, if FACTOR is used many no of times in a program and you want to change the value, all you had to do is change the EQU statement at beginning, it will changes the rest of all.

PROC (Procedure) Directive

- PROC - The PROC directive is used to identify the start of a procedure. The term near or far is used to specify the type of the procedure.
- Example:
  - SMART PROC FAR ; This identifies that the start of a procedure named as SMART and instructs the assembler that the procedure is far .
  - SMART ENDP ; This PROC is used with ENDP to indicate the break of the procedure.

ORG (ORIGIN)

- ORG Changes the starting offset address of the data in the data segment. As an assembler assembles a section of a data declarations or instruction statements, it uses a location counter to keep track of how many bytes it is from the start of a segment at any time.

- The location counter is automatically set to 0000 when assembler starts reading a segment.

- The ORG directive allows you to set the location counter to a desired value at any point in the program.

Example:

  - The statement ORG 2000H tells the assembler to set the location counter to 2000H.

SEGMENT

- SEGMENT directive : to indicate the start of a logical segment
- Syntax: Segment_name SEGMENT
- Additional terms are often added to a SEGMENT directive statement to indicate some special way in which we want the assembler to treat the segment.
  Example:
  - CODE SEGMENT WORD ; tells the assembler that we want the content of this segment located on the next available word (even address) when segments are combined and given absolute addresses.
  - Without this WORD addition, the segment will be located on the next available paragraph (16-byte) address, which might waste as much as 15 bytes of memory.
  - The statement CODE SEGMENT PUBLIC tells the assembler that the segment may be put together with other segments named CODE from other assembly modules when the modules are linked together.