Microprocessor Architecture

Internal Registers of 8086

The 8086 has four groups of the user accessible internal registers. They are the instruction pointer, four data registers, four pointer and index register, four segment registers. The 8086 has a total of fourteen 16-bit registers including a 16 bit register called the ***status register***, with 9 of bits implemented for status and control flags. Most of the registers contain data/instruction offsets within 64 KB memory segment.

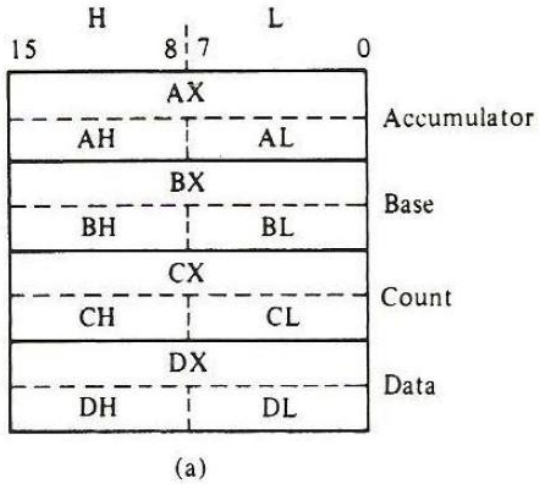
There are four different 64 KB segments for instructions, stack, data and extra data. To specify where in 1 MB of processor memory these 4 segments are located the processor uses four segment registers:

- **Code segment** (CS) is a 16-bit register containing address of 64 KB segment with processor instructions. The processor uses CS segment for all accesses to instructions referenced by instruction pointer (IP) register. CS register cannot be changed directly. The CS register is automatically updated during far jump, far call and far return instructions.
- **Stack segment (SS)** is a 16-bit register containing address of 64KB segment with program stack. By default, the processor assumes that all data referenced by the stack pointer (SP) and base pointer (BP) registers is located in the stack segment. SS register can be changed directly using POP instruction.
- **Data segment (DS)** is a 16-bit register containing address of 64KB segment with program data. By default, the processor assumes that all data referenced by general registers (AX, BX, CX, DX) and index register (SI, DI) is located in the data segment. DS register can be changed directly using POP and LDS instructions.

- **Extra segment (ES)** is a 16-bit register containing address of 64KB segment, usually with program data. By default, the processor assumes that the DI register references the ES segment in string manipulation instructions. ES register can be changed directly using POP and LES instructions. It is possible to change default segments used by general and index registers by prefixing instructions with a CS, SS, DS or ES prefix.

All general registers of the 8086 microprocessor can be used for arithmetic and logic operations.

The general registers are:



Register	Operations
AX	Word multiply, word divide, word I/O
AL	Byte multiply, byte divide, byte I/O, translate, decimal arithmetic
AH	Byte multiply, byte divide
BX	Translate
CX	String operations, loops
CL	Variable shift and rotate
DX	Word multiply, word divide, indirect I/O

(b)

Fig 1: (a) General purpose data Registers, (b) dedicated register functions

- **Accumulator** register consists of two 8-bit registers AL and AH, which can be combined together and used as a 16-bit register AX. AL in this case contains the low-order byte of the word, and AH contains the high-order byte. Accumulator can be used for I/O operations and string manipulation.
- **Base** register consists of two 8-bit registers BL and BH, which can be combined together and used as a 16-bit register BX. BL in this case contains the low-order byte of the word, and BH contains the high-order byte. BX

register usually contains a data pointer used for based, based indexed or register indirect addressing.

- **Count** register consists of two 8-bit registers CL and CH, which can be combined together and used as a 16-bit register CX. When combined, CL register contains the low-order byte of the word, and CH contains the high-order byte. Count register can be used in Loop, shift/rotate instructions and as a counter in string manipulation,.
- **Data** register consists of two 8-bit registers DL and DH, which can be combined together and used as a 16-bit register DX. When combined, DL register contains the low-order byte of the word, and DH contains the high-order byte. Data register can be used as a port number in I/O operations. In integer 32-bit multiply and divide instruction the DX register contains high-order word of the initial or resulting number.

Pointer and Index Registers

The 8086 has four other general-purpose registers**, two pointer registers SP and BP, and two index registers DI and SI.** These are used to store what are called ***offset addresses*****. An offset address represents the displacement of a storage location in memory from the segment base address in a segment register.** Unlike the general-purpose data registers, the pointer and index registers are only accessed as words (16 bits).

The stack pointer (**SP**) and base pointer (**BP**) are used with the stack segment register (**SS**) to access memory locations within the stack segment. The source index (**SI**) and destination index (**DI**) are used with **DS** or **ES** to generate addresses for instructions that access data stored in the data segment of memory.

SS: SP, BP

CS: IP

DS: SI

ES: DI

Other registers:

Instruction Pointer (IP) is a 16-bit register.

Flags is a 16-bit register containing 9 one bit flags.

- **Overflow Flag (OF)** - set if the result is too large positive number, or is too small negative number to fit into destination operand.
- **Direction Flag (DF)** - if set then string manipulation instructions will auto-decrement index registers. If cleared then the index registers will be auto-incremented.
- **Interrupt-enable Flag (IF)** - setting this bit enables maskable interrupts.
- **Single-step Flag (TF)** - if set then single-step interrupt will occur after the next instruction.
- **Sign Flag (SF)** - set if the most significant bit of the result is set.
- **Zero Flag (ZF)** - set if the result is zero
- **Auxiliary carry Flag (AF)** - set if there was a carry from or borrow to bits 0-3 in the AL register.
- **Parity Flag (PF)** - set if parity (the number of "1" bits) in the low-order byte of the result is even.
- **Carry Flag (CF)** - set if there was a carry from or borrow to the most significant bit during last result calculation.

Generating a memory address

In 8086, **logical address**isdescribed by combining two parts: **Segment address** and **offset**.

Segment address is 16-bit data from one of the segment registers (CS, SS, DS and ES).

Offset address is 16-bit data from one of the index and pointer registers (DI, SI, SP and BP). Also it could be base register BX.

To express the 20-bit **PhysicalAddress** of memory 1 Multiply Segment register by **10H** (or shift it to left by four bit)

2 Add it to the offset (see Fig 2)

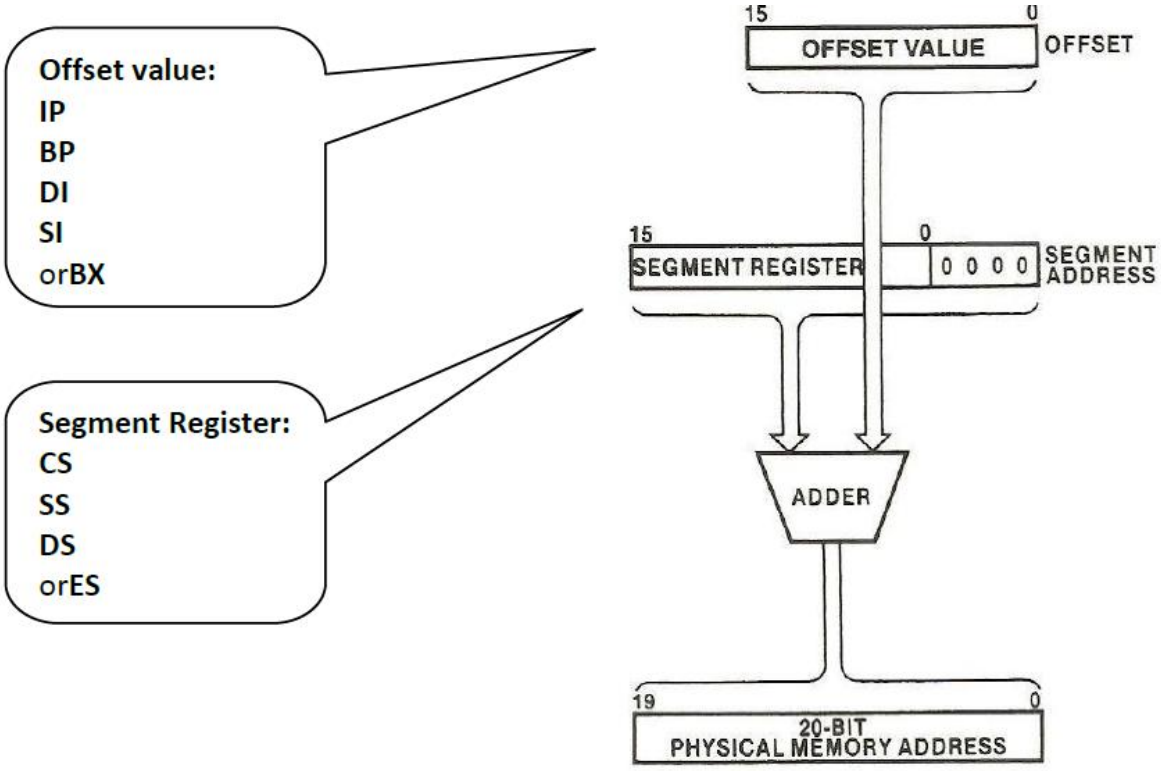


Fig (2) Generating a Memory Address