8086 Microprocessor

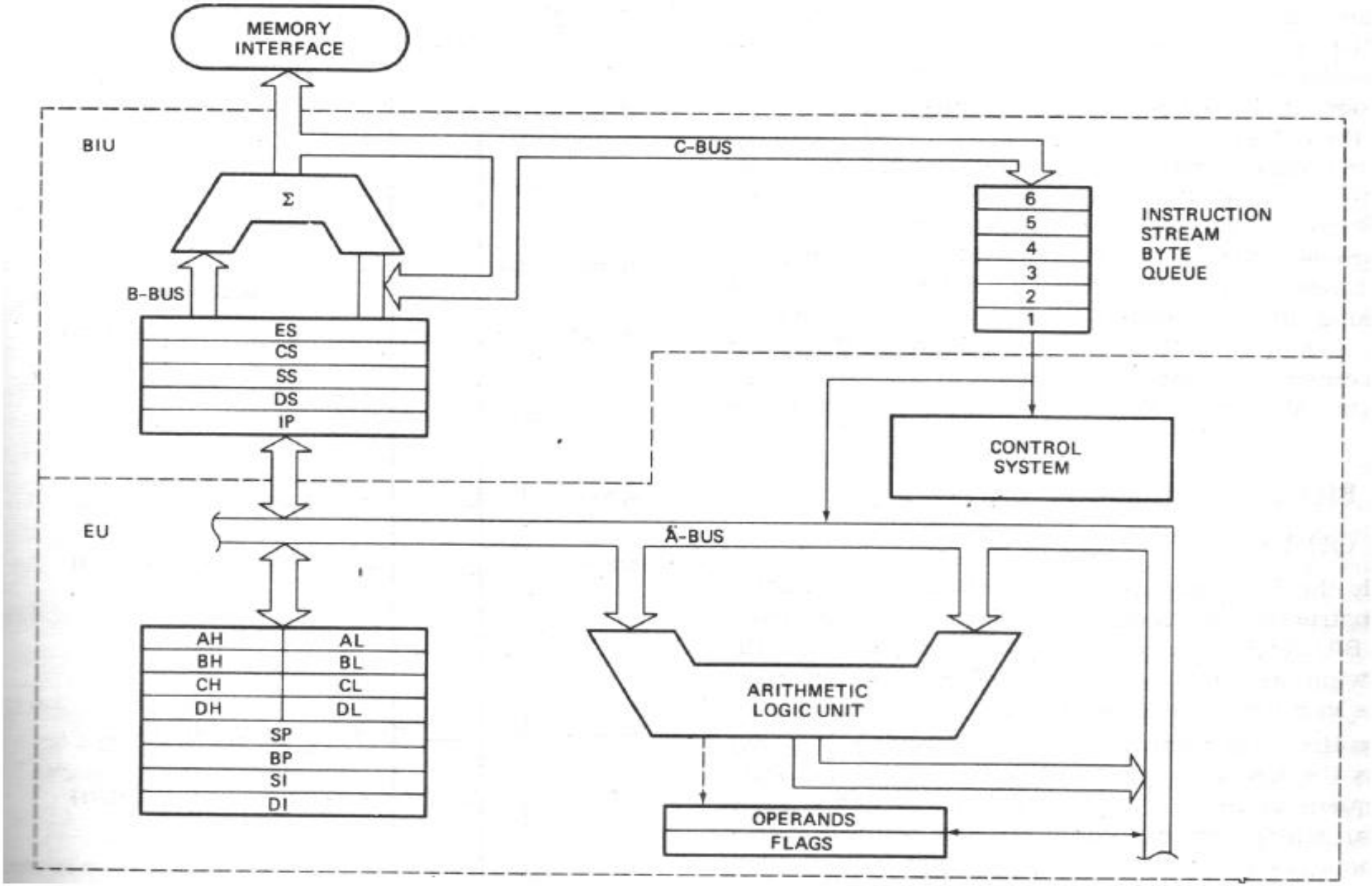
Feature of 8086 Microprocessor

- - 8086 is a 16bit processor. It’s ALU, internal registers works with 16bit binary word
- - 8086 has a 16bit data bus. It can read or write data to a memory/port either 16bits or 8 bit at a time
- - 8086 has a 20bit address bus which means, it can address upto 220 = 1MB memory location
- - Frequency range of 8086 is 6-10 MHz

Architecture

- Basic Components
- CPU Registers
  - special memory locations constructed from flip-flops and implemented on-chip
  - e.g., accumulator, count register, flag register
- Arithmetic and Logic Unit (ALU)
  - ALU is where most of the action take place inside the CPU

8086 Internal Block diagram (Intel Corp.)



Basic Components

- Bus Interface Unit (BIU)
  - responsible for controlling the address and data busses when accessing main memory and data in the cache
- Control Unit and Instruction Set
  - CPU has a fixed set of instructions to work on, e.g., MOV, CMP, JMP

Microprocessor Architecture
Instruction processing

- Modern microprocessors can process several instructions simultaneously at various stages of execution
  - this ability is called *pipelining*
- Operation of a pipelined microprocessor like the Intel 80486

Microprocessor Architecture
Instruction processing

- Processing of an instruction by microprocessor consists of three basic steps
  1. fetch instruction from the memory
  2. decode the instruction
  3. execute (usually involves accessing the memory for getting operands and storing results)
- Operation of an early processor like the Intel 8085

Important 8086 Pin Diagram/Description

- **AD15±AD0**ADDRESS DATA BUS: These lines constitute the time multiplexed memory/IO address and data bus.
- **ALE** Address Latch Enable. A HIGH on this line causes the lower order 16bit address bus to be latched that stores the addresses and then, the lower order 16bit of the address bus can be used as data bus.

- **READY** READY is the acknowledgement from the addressed memory or I/O device that it will complete the data transfer.
- **INTR** INTERRUPT REQUEST: is a level triggered input which is sampled during the last clock cycle of each instruction to determine if the processor should enter into an interrupt acknowledge operation. A subroutine is vectored to via an interrupt vector lookup table located in system memory. It can be internally masked by software resetting the interrupt enable bit. INTR is internally synchronized. This signal is active HIGH.

Registers

- Most of the registers contain data/instruction offsets within 64 KB memory segment. There are four different 64 KB segments for instructions,
- Code Segment
- stack Segment
- data Segment and
- extra data. To specify where in 1 MB of processor memory these 4 segments are located the processor uses four segment registers:

REGISTERS

- **Code segment** (CS)
- **Stack segment** (SS)
- **Data segment** (DS)
- **Extra segment** (ES)

Code segment

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, for call and for return instructions.

Stack segment

- **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

- **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

- **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.

General Register

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

Accumulator

- **Accumulator** register consists of 2 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

- **Base** register consists of 2 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

- **Count** register consists of 2 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 as a counter in string manipulation and shift/rotate instructions.

Data Register

- **Data** register consists of 2 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.

General and Index registers:

- **Stack Pointer** (SP)
- **Base Pointer** (BP)
- **Source Index** (SI)
- **Destination Index** (DI)

- **Stack Pointer** (SP) is a 16-bit register pointing to program stack.
- **Base Pointer** (BP) is a 16-bit register pointing to data in stack segment. BP register is usually used for based, based indexed or register indirect addressing.
- **Source Index** (SI) is a 16-bit register. SI is used for indexed, based indexed and register indirect addressing, as well as a source data address in string manipulation instructions.

- **Destination Index** (DI) is a 16-bit register. DI is used for indexed, based indexed and register indirect addressing, as well as a destination data address in string manipulation instructions.

Other registers

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

- **Flag Register:**

Flag is a 16-bit register containing 9 1-bit flags

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.

Flags…

- **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.

Flags

- **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.

8086 instruction set:

- **Data moving instructions.**
- **Arithmetic** - add, subtract, increment, decrement, convert byte/word and compare.
- **Logic** - AND, OR, exclusive OR, shift/rotate and test.
- **String manipulation** - load, store, move, compare and scan for byte/word.
- **Control transfer** - conditional, unconditional, call subroutine and return from subroutine.
- **Input/Output instructions**.
- Other - setting/clearing flag bits, stack operations, software interrupts, etc.

Addressing modes…

- **Implied -** the data value/data address is implicitly associated with the instruction.
- **Register -** references the data in a register or in a register pair.
- **Immediate -** the data is provided in the instruction.
- **Direct -** the instruction operand specifies the memory address where data is located.
- **Register indirect -** instruction specifies a register containing an address, where data is located. This addressing mode works with SI, DI, BX and BP registers.

Addressing modes

- **Based** - 8-bit or 16-bit instruction operand is added to the contents of a base register (BX or BP), the resulting value is a pointer to location where data resides.
- **Indexed** - 8-bit or 16-bit instruction operand is added to the contents of an index register (SI or DI), the resulting value is a pointer to location where data resides.
- **Based Indexed** - the contents of a base register (BX or BP) is added to the contents of an index register (SI or DI), the resulting value is a pointer to location where data resides.
- **Based Indexed with displacement** - 8-bit or 16-bit instruction operand is added to the contents of a base register (BX or BP) and index register (SI or DI), the resulting value is a pointer to location where data resides.