Department of Computer Engineering Techniques (Stage: 4)

Advance Computer Technologies

Dr.: Mayas Aljibawi

Change to binary

339 ---→ 101010011

512 256 128 64 32 16 8 4 2 1

339-256 = 83 -> 83 -64= 19 -> 19-16 =3 ->3-2=1 1-1=0

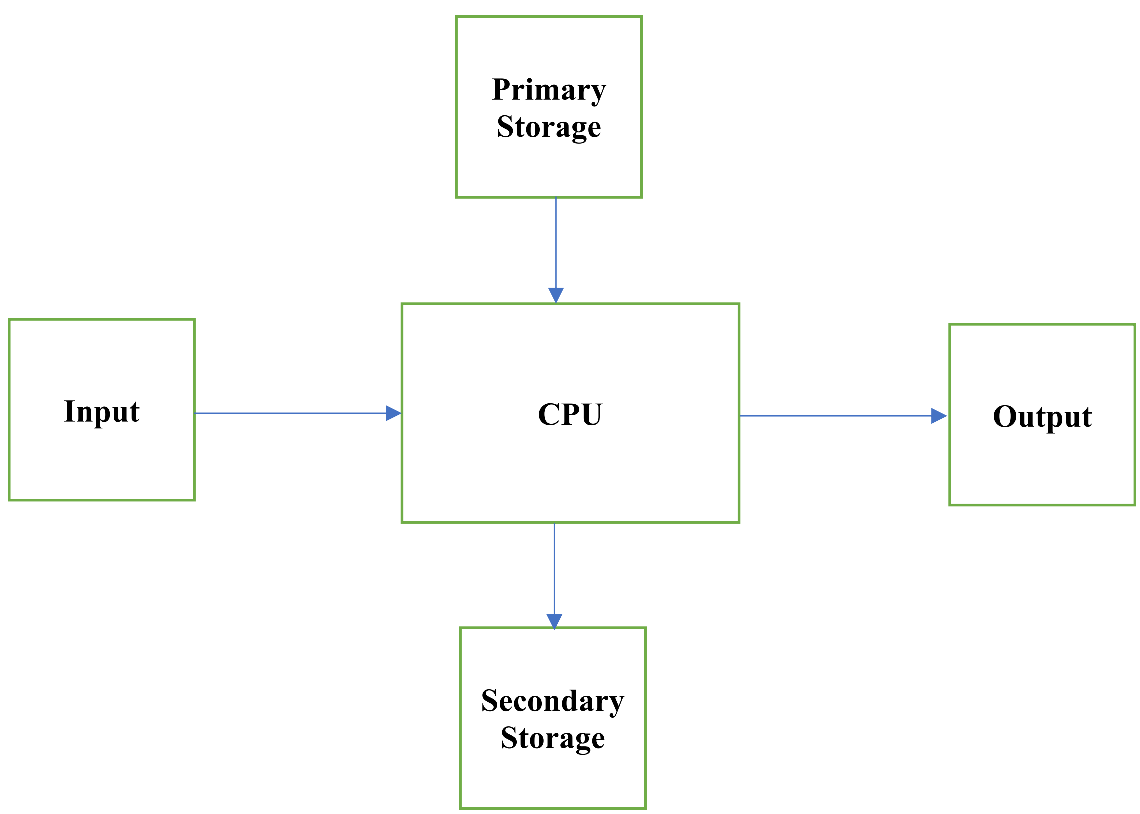
339/2 is 169 r 1 -> 169/2= 84 r 1 84/2=42 r 0 42/2=21 r 0 21/2=10 r 1 10/2=5 r 0 5/2=2 r 1 2/2=1 r 0 1/2 = 0 r 1

CPU

The central processing unit (CPU) is the most important hardware component in a computer. It has two main functions:

- to process data and instructions
- to control the rest of the computer system

All programs and data processing are run in the CPU and all hardware components are, to some extent, controlled by it.



The CPU receives instructions and data from an input or memory. The instructions and data are processed by the CPU and the results are either sent to an output or transferred to secondary storage. Data is held in primary storage while it is being processed.

Primary memory usually refers to Random Access Memory (RAM), while secondary storage refers to devices such as hard disk drives, solid state drives, removable “USB” drives, CDs, and DVDs.

The central processing unit (CPU) consists of six main components:

- control unit (CU)
- arithmetic logic unit (ALU)
- registers
- cache
- buses
- clock

Control unit (CU)

The CU provides several functions:

- it fetches, decodes and executes instructions
- it issues control signals that control hardware components within the CPU
- it transfers data and instructions around the system

Arithmetic logic unit (ALU)

The ALU has two main functions:

- it performs arithmetic and logical operations (decisions).
- it acts as a gateway between primary storage and secondary storage - data transferred between them passes through the ALU.

Registers

Registers are small amounts of high-speed memory contained within the CPU. They are used by the processor to store small amounts of data that are needed during processing, such as:

- the address of the next instruction to be executed
- the current instruction being decoded
- the results of calculations

Cache

Cache is a small amount of high-speed random access memory (RAM) built directly within the processor. It is used to temporarily hold data and instructions that the processor is likely to reuse. This allows for faster processing, as the processor does not have to wait for the data and instructions to be fetched from the RAM.

Clock

The CPU contains a clock which, along with the CU, is used to coordinate all of the computer's components. The clock sends out a regular electrical pulse which synchronises (keeps in time) all the components.

The frequency of the pulses is known as clock speed. Clock speed is measured in hertz (Hz). The greater the speed, the more instructions can be performed in any given moment of time.

In the 1980s, processors commonly ran at a rate of between 3 megahertz (MHz) and 5 MHz, which is 3 million to 5 million pulses or cycles per second. Today, processors commonly run at a rate of between 3 gigahertz (GHz) and 5 GHz, which is 3 billion to 5 billion pulses or cycles per second.

Buses

A bus is a high-speed internal connection. Buses are used to send control signals and data between the processor and other components.

Three types of bus are used.

- Address bus - carries memory addresses from the processor to other components such as primary storage and input/output devices. The address bus is unidirectional.
- Data bus - carries the data between the processor and other components. The data bus is bidirectional.
- Control bus - carries control signals from the processor to other components. The control bus also carries the clock's pulses. The control bus is unidirectional.

CISC vs. RISC

Until the early 1980s, all CPUs, whether single-chip or whole-board, followed the *CISC* (complex instruction set computer) design philosophy. CISC refers to CPUs with hundreds of instructions designed for every possible situation. To design CPUs with so many instructions consumed not only hundreds of thousands of transistors, but also made the design very complicated, time-consuming, and expensive. In the early 1980s, a new CPU design philosophy called *RISC* (reduced instruction set computer) was developed. The proponents of RISC argued that no one was using all the instructions etched into the brain of CISC-type CPUs. Why not streamline the instructions by simplifying and reducing them from hundreds to around 40 or so and use all the transistors that are saved to enhance the power of the CPU? Although the RISC concept had been explored by computer scientists at IBM as early as the 1970s, the first working single-chip RISC microprocessor was implemented by a group of researchers at the University of California at Berkeley in 1980. Today the RISC design philosophy is no longer an experiment limited to research laboratories. Since the late 1980s, many companies designing new CPUs (either single-chip or whole-board) have used the RISC philosophy. It appears that eventually the only CISC microprocessors remaining in use will be members of the 80x86 family (8086, 8088, 80286, 80386, 80486, 80586, etc.) and the 680x0 family (68000, 68010, 68020, 68030, 68040, 68050, etc.). The 80x86 will be kept alive by the huge base of IBM PC, PS, and compatible computers, and the Apple Macintosh is prolonging the life of 680x0 microprocessors.

Two main philosophies

CISC: Early computer design use CISC (complex instruction set computer)which use hundreds of instruction to every possible situation which make the design very complex , time consuming , expensive and use small set of register.

RISC: early 1980, after CISC new design call RISC (Reduced instruction set computer) was developed .The designer show the small set of instructions are use so that they reduce the number of instruction and use the all the transistor that save to enhance the power of CPU by increase number of register set and modify the control unit.

***16 bit means we can transfer 16 bits in one time.

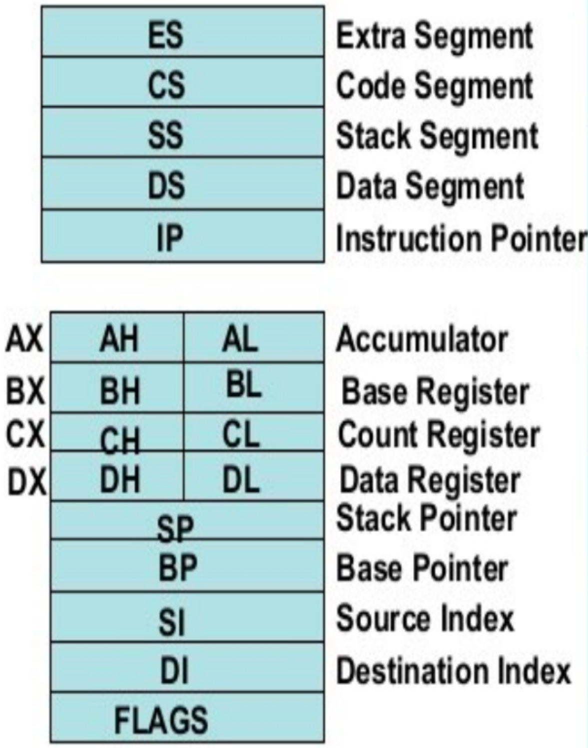
Product	8080	8085	8086	8088	80286	80386	80486
Year introduced	1974	1976	1978	1979	1982	1985	1989
Clock rate (MHz)	2 - 3	3 - 8	5 - 10	5 - 8	6 - 16	16 - 33	25 - 50
No. transistors	4500	6500	29,000	29,000	130,000	275,000	1.2 million
Physical memory	64K	64K	1M	1M	16M	4G	4G
Internal data bus	8	8	16	16	16	32	32
External data bus	8	8	16	8	16	32	32
Address bus	16	16	20	20	24	32	32
Data type (bits)	8	8	8, 16	8, 16	8, 16	8, 16, 32	8, 16, 32

INSIDE THE COMPUTER

```
Bit                                    0
Nibble                              0000
Byte                         0000   0000
Word           0000   0000   0000   0000
```

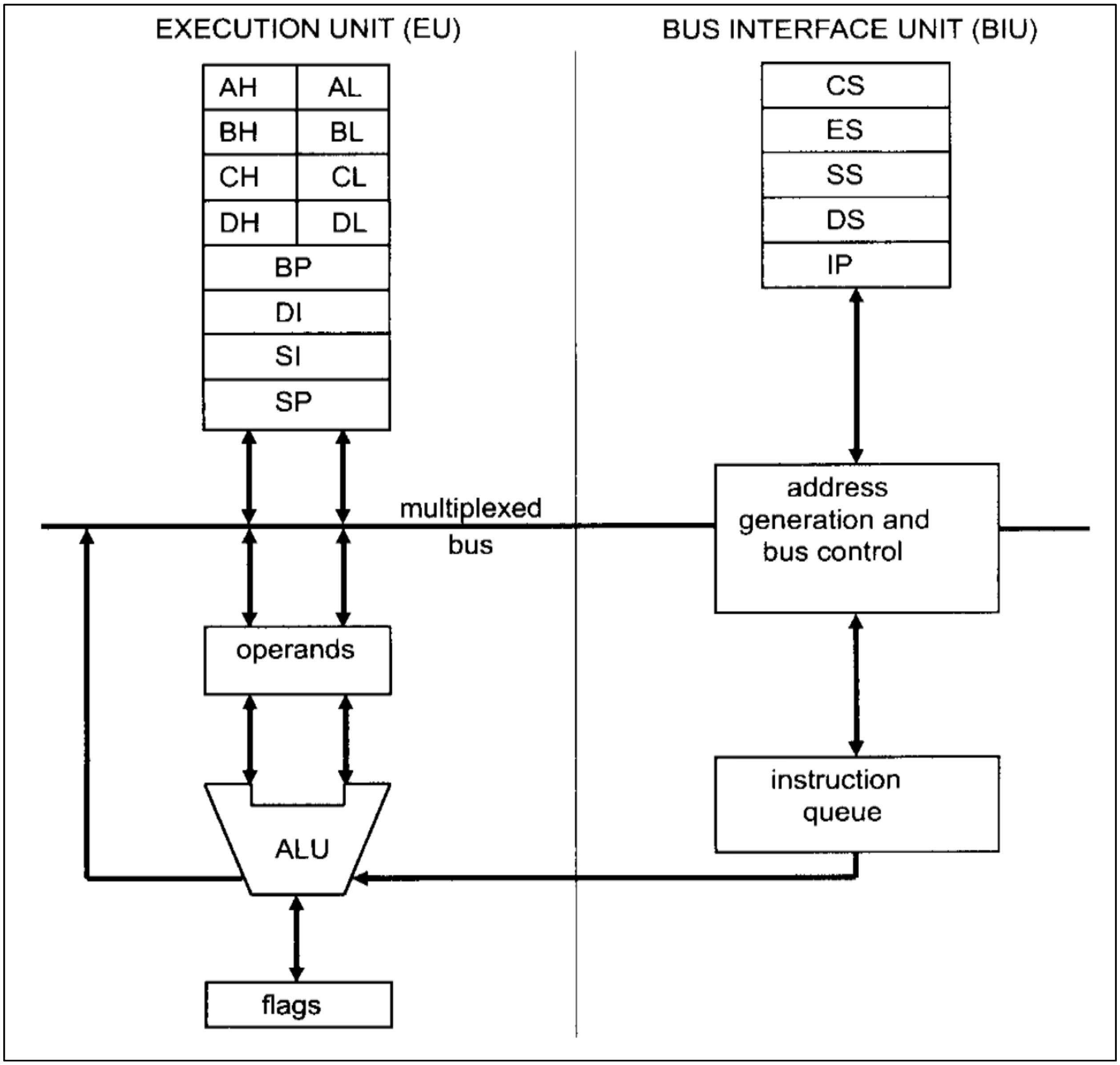
A *kilobyte* is 2^{10} bytes, which is 1024 bytes. The abbreviation K is often used. For example, some floppy disks hold 356K bytes of data. A *megabyte*, or meg as some call it, is 2^{20} bytes. That is a little over 1 million bytes; it is exactly 1,048,576. Moving rapidly up the scale in size, a *gigabyte* is 2^{30} bytes (over 1 billion), and a *terabyte* is 2^{40} bytes (over 1 trillion). As an example of how some of these terms are used, suppose that a given computer has 16 megabytes of memory. That would be 16×2^{20}, or $2^4 \times 2^{20}$, which is 2^{24}. Therefore 16 megabytes is 2^{24} bytes.

8086 REGISTER ORGANIZATION



Type	Register size	Name of the Register
General purpose registers	16 bit	AX, BX, CX, DX
	8 bit	AL, AH, BL, BH, CL, CH, DL, DH
Pointer registers	16 bit	SP, BP
Indexed registers	16 bit	SI, DI
Instruction Pointer	16 bit	IP
Segment registers	16 bit	CS, DS, SS, ES
Flags	16 bit	Flag register

Inside the 8086



EU(Execution Unit)

- EU is responsible for **program execution**
- Contains of an Arithmetic Logic Unit (ALU), a Control Unit (CU) and a number of registers

BIU (Bus Interface Unit)

- **Delivers data and instructions** to the EU.
- manage the bus control unit, segment registers and instruction queue.
- The BIU controls the buses that transfer the data to the EU, to memory and to external input/output devices, whereas the segment registers control memory addressing.

Pipelining

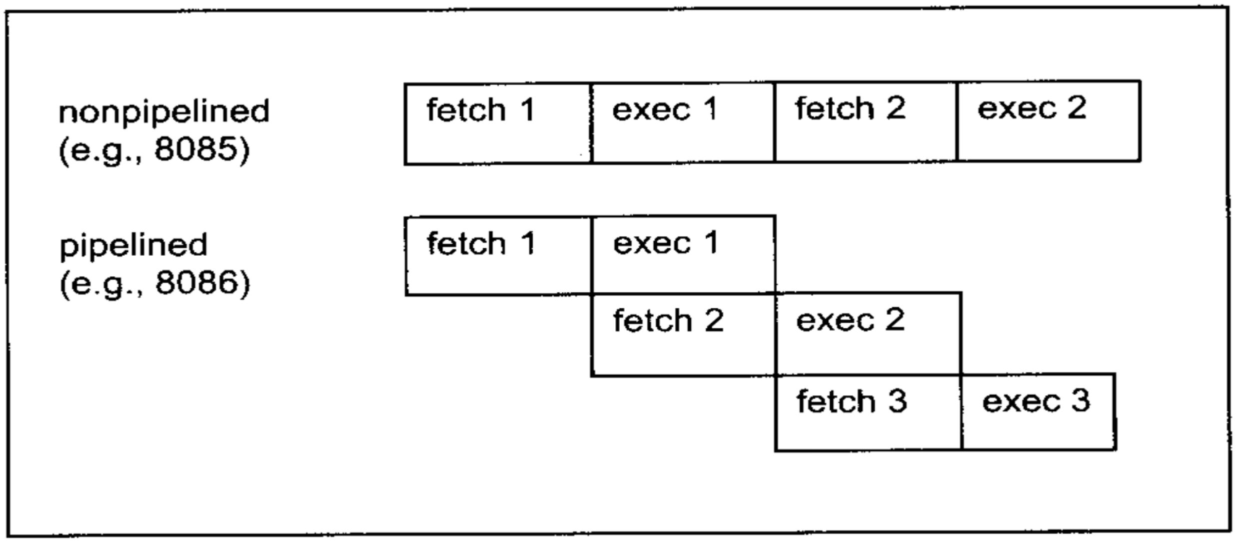


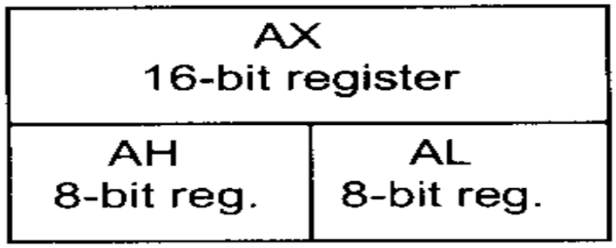
Figure 1-2. Pipelined vs. Nonpipelined Execution

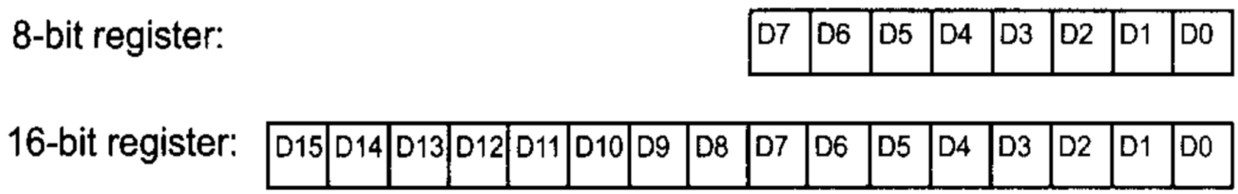
Pipelining

- 8085 has one unit only which is use in fetch and execute stage when fetch the CPU idle and when execute the bus idle.

- 8086 has two separate unit one for fetch call BIU and one for execute call EU they are connect through 6 byte queue.

Registers





The least significant byte of AX can be used as a single 8-bit register called AL, while the most significant byte of AX can be used as a single 8-bit register called AH.

Table 1-2: Registers of the 8086/286 by Category

Category	Bits	Register Names
General	16	AX, BX, CX, DX
	8	AH, AL, BH, BL, CH, CL, DH, DL
Pointer	16	SP (stack pointer), BP (base pointer)
Index	16	SI (source index), DI (destination index)
Segment	16	CS (code segment), DS (data segment), SS (stack segment), ES (extra segment)
Instruction	16	IP (instruction pointer)
Flag	16	FR (flag register)

INTRODUCTION TO ASSEMBLY PROGRAMMING

MOV instruction

Simply stated, the MOV instruction copies data from one location to another. It has the following format:

```
MOV   destination,source      ;copy source operand to destination
```

```
MOV   CL,55H     ;move 55H into register CL
MOV   DL,CL      ;copy the contents of CL into DL (now DL=CL=55H)
MOV   AH,DL      ;copy the contents of DL into AH (now AH=DL=55H)
MOV   AL,AH      ;copy the contents of AH into AL (now AL=AH=55H)
MOV   BH,CL      ;copy the contents of CL into BH (now BH=CL=55H)
MOV   CH,BH      ;copy the contents of BH into CH (now CH=BH=55H)
```

The use of 16-bit registers is demonstrated below.

```
MOV   CX,468FH   ;move 468FH into CX (now CH=46,CL=8F)
MOV   AX,CX      ;copy contents of CX to AX (now AX=CX=468FH)
MOV   DX,AX      ;copy contents of AX to DX (now DX=AX=468FH)
MOV   BX,DX      ;copy contents of DX to BX (now BX=DX=468FH)
MOV   DI,BX      ;now DI=BX=468FH
MOV   SI,DI      ;now SI=DI=468FH
MOV   DS,SI      ;now DS=SI=468FH
MOV   BP,DI      ;now BP=DI=468FH
```

In the 8086 CPU, data can be moved among all the registers shown in Table 1-2 (except the flag register) as long as the source and destination registers match in size. Code such as "MOV AL,DX" will cause an error,

If data can be moved among all registers including the segment registers, can data be moved directly into all registers? The answer is no. Data can be moved directly into nonsegment registers only, using the MOV instruction. For example,

```
MOV   AX,58FCH      ;move 58FCH into AX    (LEGAL)
MOV   DX,6678H      ;move 6678H into DX    (LEGAL)
MOV   SI,924BH      ;move 924B  into SI    (LEGAL)
MOV   BP,2459H      ;move 2459H into BP    (LEGAL)
MOV   DS,2341H      ;move 2341H into DS    (ILLEGAL)
MOV   CX,8876H      ;move 8876H into CX    (LEGAL)
MOV   CS,3F47H      ;move 3F47H into CS    (ILLEGAL)
MOV   BH,99H        ;move 99H into BH      (LEGAL)
```

From the discussion above, note the following three points:

1. Values cannot be loaded directly into any segment register (CS, DS, ES, or SS). To load a value into a segment register, first load it to a nonsegment register and then move it to the segment register, as shown next.

```
MOV   AX,2345H      ;load 2345H into AX
MOV   DS,AX         ;then load the value of AX into DS

MOV   DI,1400H      ;load 1400H into DI
MOV   ES,DI         ;then move it into ES, now ES=DI=1400
```

2. If a value less than FFH is moved into a 16-bit register, the rest of the bits are assumed to be all zeros. For example, in "MOV BX,5" the result will be BX = 0005; that is, BH = 00 and BL = 05.

3. Moving a value that is too large into a register will cause an error.

```
MOV   BL,7F2H       ;ILLEGAL: 7F2H is larger than 8 bits
MOV   AX,2FE456H    ;ILLEGAL: the value is larger than AX
```

ADD instruction

The ADD instruction has the following format:

```
ADD     destination,source          ;ADD the source operand to the destination
```

```
MOV     AL,25H          ;move 25 into AL
MOV     BL,34H          ;move 34 into BL
ADD     AL,BL           ;AL = AL + BL
```

```
MOV     DH,25H          ;move 25 into DH
MOV     CL,34H          ;move 34 into CL
ADD     DH,CL           ;add CL to DH: DH = DH + CL

MOV     DH,25H          ;load one operand into DH
ADD     DH,34H          ;add the second operand to DH

MOV     AX,34EH         ;move 34EH into AX
MOV     DX,6A5H         ;move 6A5H into DX
ADD     DX,AX           ;add AX to DX: DX = DX + AX

MOV     CX,34EH         ;load 34EH into CX
ADD     CX,6A5H         ;add 6A5H to CX (now CX=9F3H)
```

The general-purpose registers are typically used in arithmetic operations. Register AX is sometimes referred to as the accumulator.

Review Questions

1. Name three features of the 8086 that were improvements over the 8080/8085.
2. What is the major difference between 8088 and 8086 microprocessors?
3. Give the size of the address bus and physical memory capacity of the following:
   (a) 8086 (b) 80286 (c) 80386
4. The 80286 is a ______-bit microprocessor, whereas the 80386 is a ______-bit microprocessor.
5. State the major difference between the 80386 and the 80386SX.
6. List additional features introduced with the 80286 that were not present in the 8086.
7. List additional features of the 80486 that were not present in the 80386.

Review Questions

1. Write the Assembly language instruction to move value 1234H into register BX.
2. Write the Assembly language instructions to add the values 16H and ABH. Place the result in register AX.
3. No value can be moved directly into which registers?
4. What is the largest hex value that can be moved into a 16-bit register? Into an 8-bit register? What are the decimal equivalents of these hex values?