Microprocessor

Program controlled semiconductor device (IC) which fetches (from memory), decodes and executes instructions.

It is used as CPU (Central Processing Unit) in computers.

1

Why do we need to learn Microprocessors/controllers?

The microprocessor is the core of computer systems.

Nowadays many communication, digital entertainment, portable devices, are controlled by them.

A designer should know what types of components he needs, ways to reduce production costs and product reliable.

2

What is a Microprocessor?

The word comes from the combination micro and processor.

- Processor means a device that processes whatever. In this context processor means a device that processes numbers, specifically binary numbers, 0's and 1's.

  To process means to manipulate. It is a general term that describes all manipulation.
  Again in this content, it means to perform certain operations on the numbers that depend on the microprocessor's design.

3

What is Micro?

Micro is a new addition.

- In the late 1960's, processors were built using discrete elements.
  These devices performed the required operation, but were too large and too slow.

- In the early 1970's the microchip was invented. All of the components that made up the processor were now placed on a single piece of silicon. The size became several thousand times smaller and the speed became several hundred times faster. The "Micro" Processor was born.

4

Definition of a Microprocessor

The microprocessor is a programmable device that takes in numbers, performs on them arithmetic or logical operations according to the program stored in memory and then produces other numbers as a result.

5

Definition Continued...

Lets expand each of the underlined words:

Programmable device: The microprocessor can perform different sets of operations on the data it receives depending on the sequence of instructions supplied in the given program.

By changing the program, the microprocessor manipulates the data in different ways.

Instructions: Each microprocessor is designed to execute a specific group of operations. This group of operations is called an instruction set. This instruction set defines what the microprocessor can and cannot do.

Takes in: The data that the microprocessor manipulates must come from somewhere.

It comes from what is called “input devices”.

These are devices that bring data into the system from the outside world. These represent devices such as a keyboard, a mouse, switches etc.

6

Definition Continued...

Numbers: The microprocessor has a very narrow view on life. It only understands binary numbers.

A binary digit is called a bit (which comes from binary digit).

The microprocessor recognizes and processes a group of bits together. This group of bits is called a "word".

The number of bits in a Microprocessor's word, is a measure of its "abilities".

7

Definition Continued.....

Words, Bytes, etc.

The earliest microprocessor (the Intel 8088 and Motorola's 6800) recognized 8-bit words.

They processed information 8-bits at a time. That's why they are called "8-bit processors". They can handle large numbers, but in order to process these numbers, they broke them into 8-bit pieces and processed each group of 8-bits separately.

Later microprocessors (8086 and 68000) were designed with 16-bit words.

A group of 8-bits were referred to as a "half-word" or "byte".
A group of 4 bits is called a "nibble".
Also, 32 bit groups were given the name "long word".

Today, all processors manipulate at least 64 bits at a time.

8

Definition Continued.....

Arithmetic and Logic Operations:

Every microprocessor has arithmetic operations such as add and subtract as part of its instruction set.

Most microprocessors will have operations such as multiply and divide. Some of the newer ones will have complex operations such as square root.

In addition, microprocessors have logic operations as well. Such as AND, OR, XOR, shift left, shift right, etc.

Again, the number and types of operations define the microprocessor's instruction set and depends on the specific microprocessor.

9

Definition Continued

Program: A program is a sequence of instructions that bring data into the microprocessor, processes it and sends it out.

There are many programming languages (C, C++, FORTRAN, and JAVA…) However, these programming languages can be grouped into three main levels (these days a fourth level is developing).

10

Definition Continued

Programming Languages

Machine language

Machine language is the lowest level programming language. It is a language intended to be understood by the microprocessor (the machine) only.
In this language, every instruction is described by binary patterns.

e.g. 11001101 may mean 1 + 2

This is the form in which instructions are stored in memory. This is the only form that the microprocessor understands.

11

Definition Continued

Programming Languages

Assembly language

This language is more understandable by humans. In this language, the binary patterns are assigned mnemonics (short abbreviated names).

e.g. “Add 1,2” is assigned to the machine language pattern 11001101 mentioned above to refer to the operation 1+2.

There is usually one assembly language instruction for each machine language instruction.

12

Definition Continued

Programming Languages

High level languages

These are languages like C, PASCAL and FORTRON. These are more natural for humans to use than assembly or machine languages. They are also more compact (i.e. it takes less statements to write the program).

One high level instruction translates into many assembly or machine language instructions.

e.g. x = y + z may translate into:

```
MOV     1000, R1
MOV     1004, R2
ADD     R1, R2
MOV     R1, 1008
```

13

Definition Continued

Programming Languages

The new level being developed: is ultra high level languages which would contain things like C++, and JAVA.

Here a single instruction may translate into hundreds of assembly or machine language instructions.

14

Definition Continued

Stored in memory :

First, what is memory?

Memory is the location where information is kept while not in current use.

Memory is a collection of storage devices. Usually, each storage device holds one bit. Also, in most kinds of memory, these storage devices are grouped into groups of 8. These 8 storage locations can only be accessed together. So, one can only read or write in terms of bytes to and form memory.

Memory is usually measured by the number of bytes it can hold. It is measured in Kilos, Megas and lately Gigas. A Kilo in computer language is 2^{10} =1024. So, a KB (KiloByte) is 1024 bytes. Mega is 1024 Kilos and Giga is 1024 Mega.

15

Definition Continued

Stored in memory:

When a program is entered into a computer, it is stored in memory. Then as the microprocessor starts to execute the instructions, it brings the instructions from memory one at a time.

Memory is also used to hold the data.

The microprocessor reads (brings in) the data from memory when it needs it and writes (stores) the results into memory when it is done.

16

Definition Continued

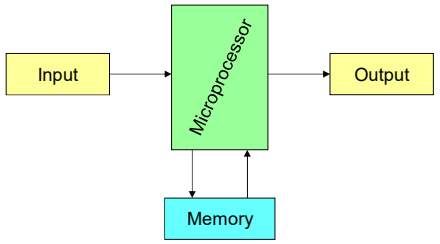
Produces: For the user to see the result of the execution of the program, the results must be presented in a human readable form.

The results must be presented on an output device.

This can be the monitor, a paper from the printer, a simple LED or many other forms.

17

From the above description, we can draw the following block diagram to represent a microprocessor-based system



18

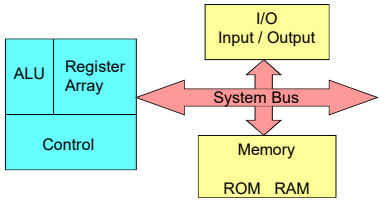
Inside the Microprocessor.

Internally, the microprocessor is made up of 3 main units.

- The Arithmetic/Logic Unit (ALU)
- The Control Unit.
- An array of registers for holding data while it is being manipulated.

19

Organization of a Microprocessor based system



20

Organization of the Microprocessor

The microprocessor can be divided into three main pieces:

- Arithmetic/Logic Unit
  - Performs all computing and logic operations such as addition and subtraction as well as AND, OR and XOR.
- Register Array
  - A collection of registers within the microprocessor itself. These are used primarily for data storage during program execution. The number and the size of these registers differ from one microprocessor to the other.
- Control Unit
  - As the name implies, the control Unit controls what is happening in the microprocessor. It provides the necessary control and timing signals to all operations in the microprocessor as well as its contact to the outside world.

21

Intruduction

Microprocessor (uP)(MPU)
- A uP is a CPU on a single chip.
- Components of CPU
  - ALU, instruction decoder, registers, bus control circuit, etc.

Micro-computer (u-Computer)
- small computer
- uP + peripheral I/O + memory specifically for data acquisition and control applications

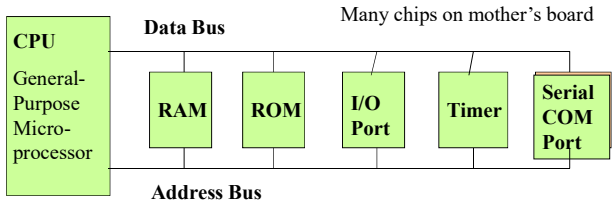
Microcontroller (uC)
- u-Computer on a single chip of silicon

22

Microprocessor:

General-purpose microprocessor

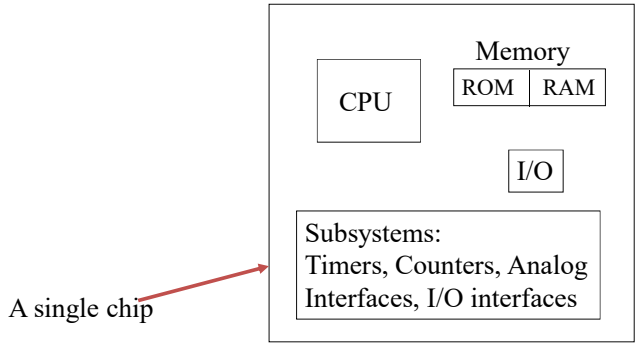
CPU for Computers
No RAM, ROM, I/O on CPU chip itself
Example : Intel's x86, Motorola's 680x0



General-Purpose Microprocessor System

23

Microcontroller



24

uP vs. uC

A uP
- only is a single-chip CPU
- bus is available
- RAM capacity, num of port is selectable
- RAM is larger than ROM (usually)

A uC
- contains a CPU and RAM,ROM , Peripherals, I/O port in a single IC
- internal hardware is fixed
- Communicate by port
- ROM is larger than RAM (usually)
- Small power consumption
- Single chip, small board
- Implementation is easy
- Low cost

25

Microprocessor vs. Microcontroller

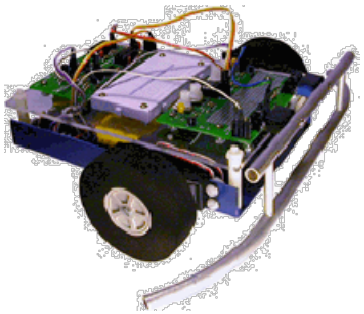
Microprocessor	Microcontroller
CPU is stand-alone, RAM, ROM, I/O, timer are separate	• CPU, RAM, ROM, I/O and timer are all on a single chip
designer can decide on the amount of ROM, RAM and I/O ports.	• fix amount of on-chip ROM, RAM, I/O ports
expansive	• for applications in which cost, power and space are critical
versatility	• single-purpose
general-purpose	

26

uP vs. uC – cont.

Applications

- uCs are suitable to control of I/O devices in designs requiring a minimum component
- uPs are suitable to processing information in computer systems.



27

uP vs. uC – cont.

uC is easy to use and design.

- Only single chip can be a complete system interfacing to other devices,
  - for example, motors, displays, sensors, and communicate with PC.

In contrast, similar system that builds from uP would require a lot of additional units,

- such as RAM, UART, I/O , TIMER and etc.

28

The Microprocessor (MPU)

The uP is the 'brain of the microcomputer'
Is a single chip which is capable of
- processing data
- controlling all of the components which make up the microcomputer system

µP used to sequence executions of instructions that is in memory
uP Fetch , Decode , and Execute the instruction
The internal architecture of the microprocessor is complex.

29

Microprocessors

Microprocessors come in all kinds of varieties from the very simple to the very complex
Depend on data bus and register and ALU width uP could be 4-bit , 8-bit , 16-bit, 32-bit , 64-bit

All uPs have
- the address bus
- the data bus
- RD, WR, CLK , RST, INT, . . .

30

What's a Chip?

A chip is also called an integrated circuit.
Generally it is a small, thin piece of silicon onto which the transistors making up the microprocessor have been etched.
A chip might be as large as an inch on a side and can contain tens of millions of transistors.
Simpler processors might consist of a few thousand transistors etched onto a chip just a few millimeters square.

31

HISTORY OF MICROPROCESSORS

1

32

INTRODUCTION

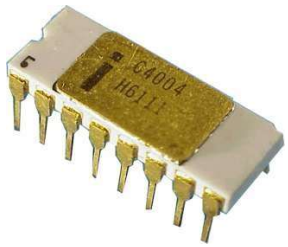
- Fairchild Semiconductors (founded in 1957) invented the first IC in 1959.
- In 1968, **Robert Noyce, Gordan Moore, Andrew Grove** resigned from Fairchild Semiconductors.
- They founded their own company **Intel** (Integrated Electronics).
- Intel grown from 3 man start-up in 1968

33

4-BIT MICROPROCESSORS

34

INTEL 4004



- Introduced in 1971.
- It was the first microprocessor by Intel.
- It was a 4-bit µP.
- Its clock speed was 740KHz.
- It had 2,300 transistors.
- It could execute around 60,000 instructions per second.

35

INTEL 4040

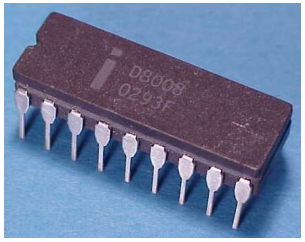
- Introduced in 1974.
- It was also 4-bit µP.

36

8-BIT MICROPROCESSORS

37

INTEL 8008



- Introduced in 1972.
- It was first 8-bit µP.
- Its clock speed was 500 KHz.
- Could execute 50,000 instructions per second.

38

INTEL 8080

- Introduced in 1974.
- It was also 8-bit µP.
- Its clock speed was 2 MHz.
- It had 6,000 transistors.
- Was 10 times faster than 8008.
- Could execute 5,00,000 instructions per second.

39

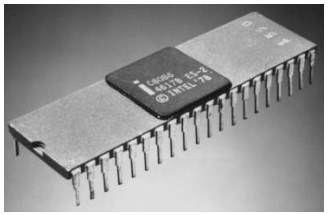
INTEL 8085

- Introduced in 1976.
- It was also 8-bit µP.
- Its clock speed was 3 MHz.
- Its data bus is 8-bit and address bus is 16-bit.
- It had 6,500 transistors.
- Could execute 7,69,230 instructions per second.
- It could access 64 KB of memory.
- It had 246 instructions.

40

16-BIT MICROPROCESSORS

INTEL 8086



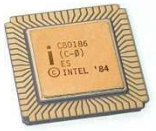
- **Introduced in 1978.**
- It was first 16-bit µP.
- Its clock speed is 4.77 MHz, 8 MHz and 10 MHz, depending on the version.
- Its data bus is 16-bit and address bus is 20-bit.
- It had 29,000 transistors.
- Could execute 2.5 million instructions per second.
- It could access 1 MB of memory.
- It had 22,000 instructions.
- It had ***Multiply*** and ***Divide*** instructions.

INTEL 8088

- Introduced in 1979.
- It was also 16-bit µP.
- It was created as a cheaper version of Intel's 8086.
- It was a 16-bit processor with an 8-bit external bus.

43

INTEL 80186 & 80188



- Introduced in 1982.
- They were 16-bit µPs.
- Clock speed was 6 MHz.

44

INTEL 80286

- Introduced in 1982.
- It was 16-bit μP.
- Its clock speed was 8 MHz.

45

32-BIT MICROPROCESSORS

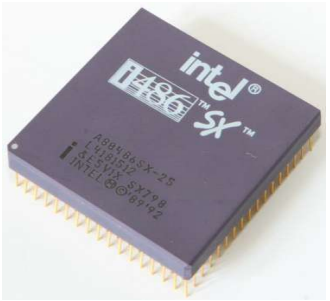
46

INTEL 80386

- **Introduced in 1986.**
- It was first 32-bit µP.
- Its data bus is 32-bit and address bus is 32-bit.
- It could address 4 GB of memory.

47

INTEL 80486



- **Introduced in 1989.**
- It was also 32-bit µP.
- It had 1.2 million transistors.
- Its clock speed varied from 16 MHz to 100 MHz depending upon the various versions.

48

INTEL PENTIUM

- **Introduced in 1993.**
- It was also 32-bit µP.
- It was originally named 80586.
- Its clock speed was 66 MHz.

49

INTEL PENTIUM PRO

- Introduced in 1995.
- It was also 32-bit µP.

50

Intel Pentium II

- Introduced in 1997.
- It was also 32-bit µP.

51

Intel Pentium II Xeon

- Introduced in 1998.
- It was also 32-bit µP.

52

Intel Pentium III

- Introduced in 1999.
- It was also 32-bit µP.

53

Intel Pentium IV

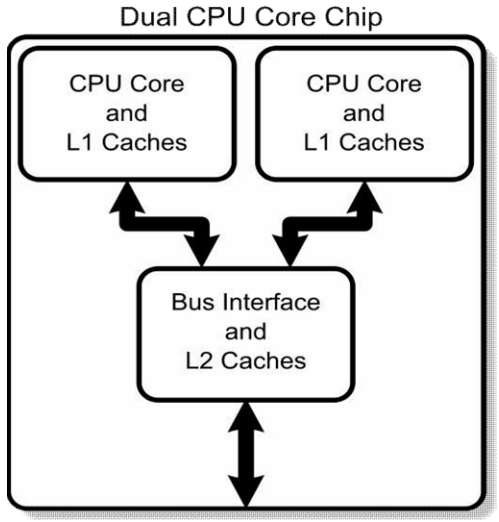
- Introduced in 2000.
- It was also 32-bit µP.

54

Intel Dual Core

- Introduced in 2006.
- It is 32-bit or 64-bit µP.
- It has two cores.
- Both the cores have there own internal bus and L1 cache, but share the external bus and L2 cache

55



56

64-BIT MICROPROCESSORS

INTEL CORE 2

- Introduced in 2006.
- It is a 64-bit µP.

Intel Core i7

- Introduced in 2008.
- It is a 64-bit µP.

59

Intel Core i5

- Introduced in 2009.
- It is a 64-bit µP.

60

Intel Core i3

- Introduced in 2010.
- It is a 64-bit µP.

61

Name	Date	Transistors	Microns	Clock speed	Data width	MIPS
8080	1974	6,000	6	2 MHz	8 bits	0.64
8088	1979	29,000	3	5 MHz	16 bits 8-bit bus	0.33
80286	1982	134,000	1.5	6 MHz	16 bits	1
80386	1985	275,000	1.5	16 MHz	32 bits	5
80486	1989	1,200,000	1	25 MHz	32 bits	20
Pentium	1993	3,100,000	0.8	60 MHz	32 bits 64-bit bus	100
Pentium II	1997	7,500,000	0.35	233 MHz	32 bits 64-bit bus	~300
Pentium III	1999	9,500,000	0.25	450 MHz	32 bits 64-bit bus	~510
Pentium 4	2000	42,000,000	0.18	1.5 GHz	32 bits 64-bit bus	~1,700
Pentium 4 ""	2004	125,000,000	0.09	3.6 GHz	32 bits 64-bit bus	~7,000

62

Information about this table:

The date is the year that the processor was first introduced. Many processors are re-introduced at higher clock speeds for many years after the original release date.

Transistors is the number of transistors on the chip. You can see that the number of transistors on a single chip has risen steadily over the years.

Microns is the width, in microns, of the smallest wire on the chip. For comparison, a human hair is 100 microns thick. As the feature size on the chip goes down, the number of transistors rises.

Clock speed is the maximum rate that the chip can be clocked at. Clock speed will make more sense in the next section.

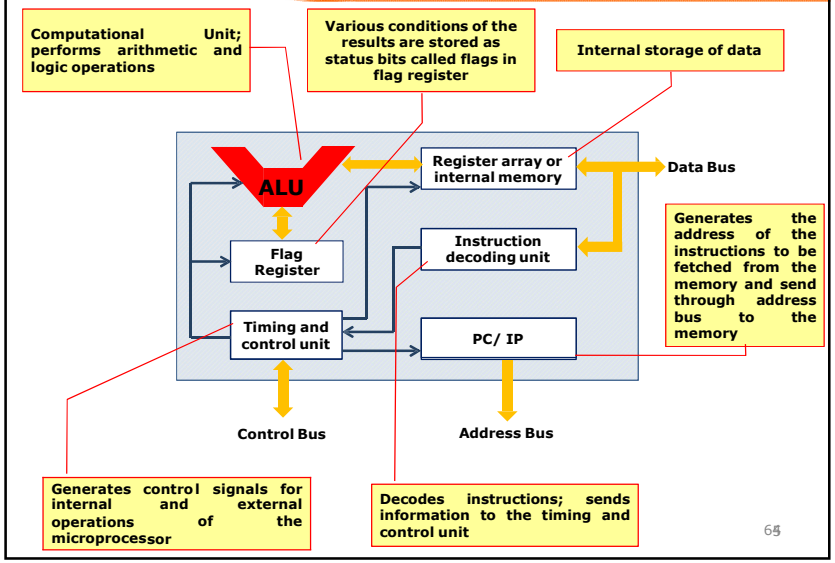
Data Width is the width of the ALU. An 8-bit ALU can add/subtract/multiply/etc. two 8-bit numbers, while a 32-bit ALU can manipulate 32-bit numbers. An 8-bit ALU would have to execute four instructions to add two 32-bit numbers, while a 32-bit ALU can do it in one instruction. In many cases, the external data bus is the same width as the ALU, but not always. The 8088 had a 16-bit ALU and an 8-bit bus, while the modern Pentiums fetch data 64 bits at a time for their 32-bit ALUs.

MIPS stands for "millions of instructions per second" and is a rough measure of the performance of a CPU. Modern CPUs can do so many different things that MIPS ratings lose a lot of their meaning, but you can get a general sense of the relative power of the CPUs from this column.

63

Microprocessor

Functional blocks



64

Microprocessor

Fifth Generation Pentium

Fourth Generation
During 1980s
Low power version of HMOS technology (HCMOS)
32 bit processors
Physical memory space 2^{24} bytes = 16 Mb
Virtual memory space 2^{40} bytes = 1 Tb
Floating point hardware
Supports increased number of addressing modes

Intel 80386

Third Generation
During 1978
HMOS technology ⇒ Faster speed, Higher packing density
16 bit processors ⇒ 40/ 48/ 64 pins
Easier to program
Dynamically relatable programs
Processor has multiply/ divide arithmetic hardware
More powerful interrupt handling capabilities
Flexible I/O port addressing

Intel 8086 (16 bit processor)

Second Generation
During 1973
NMOS technology ⇒ Faster speed, Higher density, Compatible with TTL
4 / 8/ 16 bit processors ⇒ 40 pins
Ability to address large memory spaces and I/O ports
Greater number of levels of subroutine nesting
Better interrupt handling capabilities

Intel 8085 (8 bit processor)

First Generation
Between 1971 – 1973
PMOS technology, non compatible with TTL
4 bit processors ⇒ 16 pins
8 and 16 bit processors ⇒ 40 pins
Due to limitations of pins, signals are multiplexed

8086 Microprocessor

Overview

First 16- bit processor released by INTEL in the year 1978

Originally HMOS, now manufactured using HMOS III technique

Approximately 29, 000 transistors, 40 pin DIP, 5V supply

Does not have internal clock; external asymmetric clock source with 33% duty cycle

20-bit address to access memory can address up to 2^{20} = 1 megabytes of memory space.

Addressable memory space is organized in to two banks of 512 kb each; Even (or lower) bank and Odd (or higher) bank. Address line A_0 is used to select even bank and control signal $\overline{BHE}$ is used to access odd bank

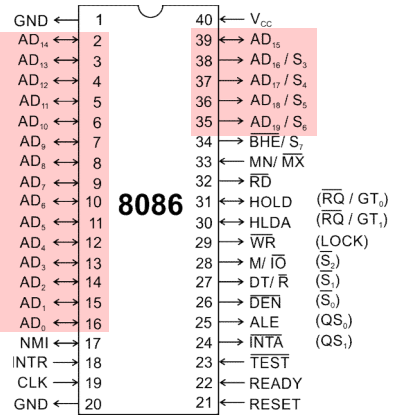
Uses a separate 16 bit address for I/O mapped devices ⇒ can generate 2^{16} = 64 k addresses.

Operates in two modes: minimum mode and maximum mode, decided by the signal at MN and $\overline{MX}$ pins.

Pins and signals 8086

Pins and Signals

Common signals



AD_0-AD_{15} (Bidirectional)

Address/Data bus

Low order address bus; these are multiplexed with data.

When AD lines are used to transmit memory address the symbol A is used instead of AD, for example A_0-A_{15}.

When data are transmitted over AD lines the symbol D is used in place of AD, for example D_0-D_7, D_8-D_{15} or D_0-D_{15}.

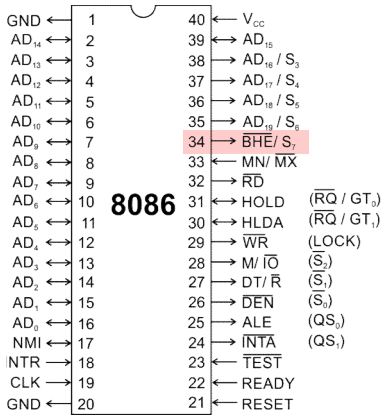
A_{16}/S_3, A_{17}/S_4, A_{18}/S_5, A_{19}/S_6

High order address bus. These are multiplexed with status signals

8086 Microprocessor

Pins and Signals

Common signals



BHE (Active Low)/S_7 (Output)

Bus High Enable/Status

It is used to enable data onto the most significant half of data bus, D_8-D_{15}. 8-bit device connected to upper half of the data bus use BHE (Active Low) signal. It is multiplexed with status signal S_7.

MN/ MX

MINIMUM / MAXIMUM

This pin signal indicates what mode the processor is to operate in.

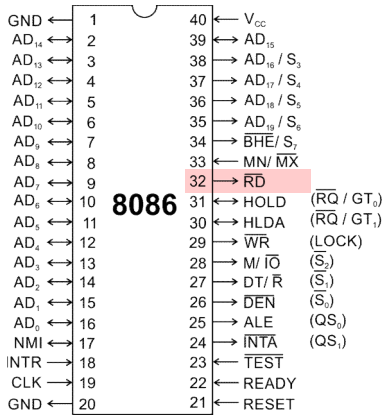
RD (Read) (Active Low)

The signal is used for read operation.
It is an output signal.
It is active when low.

8086 Microprocessor

Pins and Signals

Common signals



TEST

TEST input is tested by the 'WAIT' instruction.

8086 will enter a wait state after execution of the WAIT instruction and will resume execution only when the TEST is made low by an active hardware.

This is used to synchronize an external activity to the processor internal operation.

READY

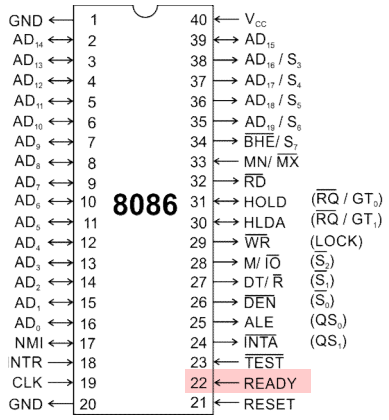
This is the acknowledgement from the slow device or memory that they have completed the data transfer.

The signal made available by the devices is synchronized by the 8284A clock generator to provide ready input to the 8086.

The signal is active high.

8086 Microprocessor

Pins and Signals — Common signals



RESET (Input)

Causes the processor to immediately terminate its present activity.

The signal must be active HIGH for at least four clock cycles.

CLK

The clock input provides the basic timing for processor operation and bus control activity. Its an asymmetric square wave with 33% duty cycle.

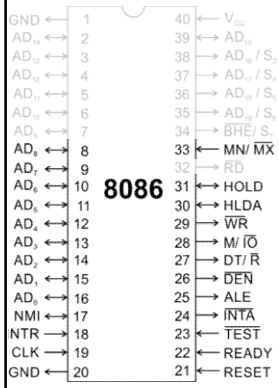
INTR Interrupt Request

This is a triggered input. This is sampled during the last clock cycles of each instruction to determine the availability of the request. If any interrupt request is pending, the processor enters the interrupt acknowledge cycle.

This signal is active high and internally synchronized.

8086 Microprocessor

Pins and Signals — Min/ Max Pins



The 8086 microprocessor can work in two modes of operations : Minimum mode and Maximum mode.

In the minimum mode of operation the microprocessor do not associate with any co-processors and can not be used for multiprocessor systems.

In the maximum mode the 8086 can work in multi-processor or co-processor configuration.

Minimum or maximum mode operations are decided by the pin MN/ MX(Active low).

When this pin is high 8086 operates in minimum mode otherwise it operates in Maximum mode.

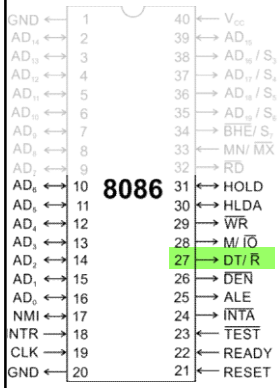
8086 Microprocessor

Pins and Signals

Minimum mode signals

For minimum mode operation, the MN/ $\overline{MX}$ is tied to VCC (logic high)

8086 itself generates all the bus control signals



DT/R — (**Data Transmit/ Receive**) Output signal from the processor to control the direction of data flow through the data transceivers

DEN — (**Data Enable**) Output signal from the processor used as out put enable for the transceivers

ALE — (**Address Latch Enable**) Used to demultiplex the address and data lines using external latches

M/IO — Used to differentiate memory access and I/O access. For memory reference instructions, it is **high**. For IN and OUT instructions, it is **low**.

WR — Write control signal; asserted **low** Whenever processor writes data to memory or I/O port

INTA — (**Interrupt Acknowledge**) When the interrupt request is accepted by the processor, the output is **low** on this line.

14

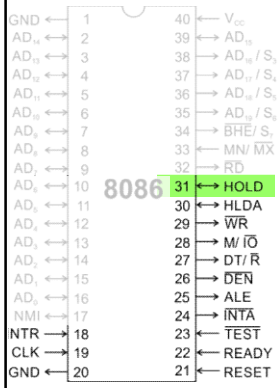
8086 Microprocessor

Pins and Signals

Minimum mode signals

For minimum mode operation, the MN/ $\overline{MX}$ is tied to VCC (logic high)

8086 itself generates all the bus control signals



HOLD — Input signal to the processor form the bus masters as a request to grant the control of the bus.

Usually used by the DMA controller to get the control of the bus.

HLDA — (**Hold Acknowledge**) Acknowledge signal by the processor to the bus master requesting the control of the bus through HOLD.

The acknowledge is asserted high, when the processor accepts HOLD.

15

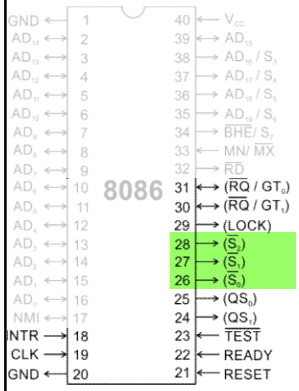
8086 Microprocessor

Pins and Signals

Maximum mode signals

During maximum mode operation, the MN/ $\overline{MX}$ is grounded (logic low)

Pins 24 -31 are reassigned



S_0, S_1, S_2 **Status signals; used by the 8086 bus controller to generate bus timing and control signals. These are decoded as shown.**

Status Signal			Machine Cycle
$\overline{S}_2$	$\overline{S}_1$	$\overline{S}_0$	
0	0	0	Interrupt acknowledge
0	0	1	Read I/O port
0	1	0	Write I/O port
0	1	1	Halt
1	0	0	Code access
1	0	1	Read memory
1	1	0	Write memory
1	1	1	Passive/Inactive

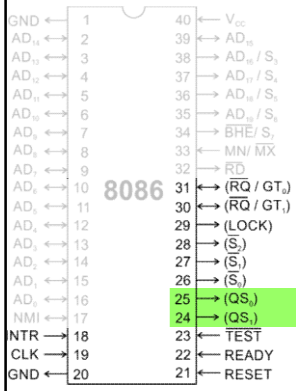
8086 Microprocessor

Pins and Signals

Maximum mode signals

During maximum mode operation, the MN/ $\overline{MX}$ is grounded (logic low)

Pins 24 -31 are reassigned



QS_0, QS_1 **(Queue Status) The processor provides the status of queue in these lines.**

The queue status can be used by external device to track the internal status of the queue in 8086.

The output on QS_0 and QS_1 can be interpreted as shown in the table.

Queue status		Queue operation
QS_1	QS_0	
0	0	No operation
0	1	First byte of an opcode from queue
1	0	Empty the queue
1	1	Subsequent byte from queue

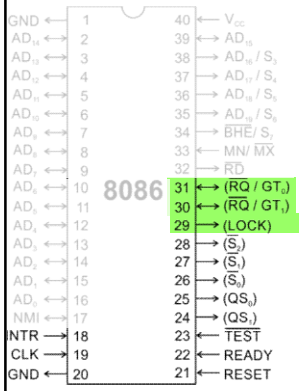
8086 Microprocessor

Pins and Signals

Maximum mode signals

During maximum mode operation, the MN/ $\overline{MX}$ is grounded (logic low)

Pins 24 -31 are reassigned



$\overline{RQ}/\overline{GT_0}$, $\overline{RQ}/\overline{GT_1}$ (Bus Request/ Bus Grant) These requests are used by other local bus masters to force the processor to release the local bus at the end of the processor's current bus cycle.

Remains active until the completion of the instruction prefixed by LOCK.

The 8086 output low on the $\overline{LOCK}$ pin while executing an instruction prefixed by LOCK to prevent other bus masters from gaining control of the system bus.

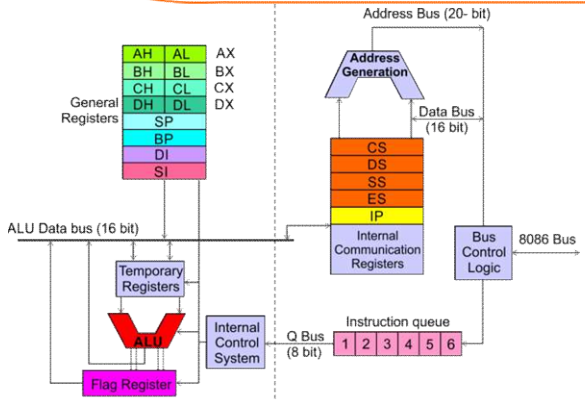
18

Architecture of 8086

78

8086 Microprocessor

Architecture



Execution Unit (EU)

EU executes instructions that have already been fetched by the BIU.

BIU and EU functions separately.

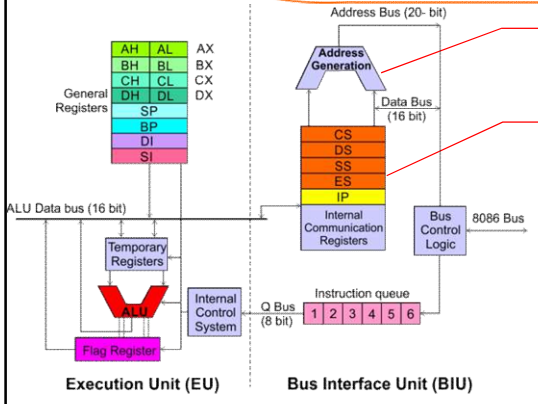
Bus Interface Unit (BIU)

BIU fetches instructions, reads data from memory and I/O ports, writes data to memory and I/ O ports.

8086 Microprocessor

Architecture

Bus Interface Unit (BIU)

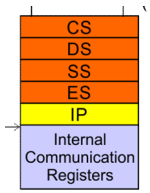


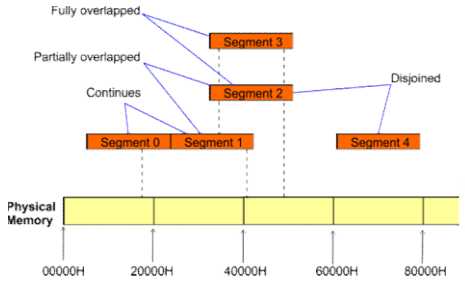
Dedicated Adder to generate 20 bit address

Four 16-bit segment registers

Code Segment (CS)
Data Segment (DS)
Stack Segment (SS)
Extra Segment (ES)

Segment Registers



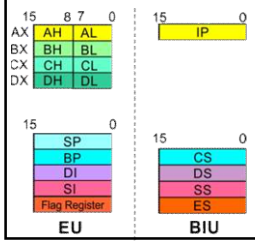


- 8086"s 1-megabyte memory is divided into segments of up to 64K bytes each.
- The 8086 can directly address four segments (256 K bytes within the 1 M byte of memory) at a particular time.
- Programs obtain access to code and data in the segments by changing the segment register content to point to the desired segments.

Segment Registers

Code Segment Register

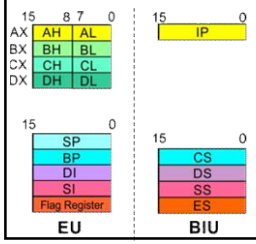
- 16-bit
- CS contains the base or start of the current code segment; IP contains the distance or offset from this address to the next instruction byte to be fetched.
- BIU computes the 20-bit physical address by logically shifting the contents of CS 4-bits to the left and then adding the 16-bit contents of IP.
- That is, all instructions of a program are relative to the contents of the CS register multiplied by 16 and then offset is added provided by the IP.



Segment Registers

Data Segment Register

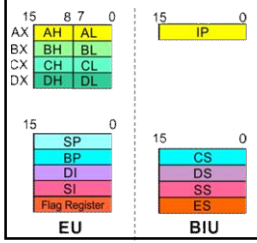
- **16-bit**
- **Points to the current data segment; operands for most instructions are fetched from this segment.**
- **The 16-bit contents of the Source Index (SI) or Destination Index (DI) or a 16-bit displacement are used as offset for computing the 20-bit physical address.**



Segment Registers

Stack Segment Register

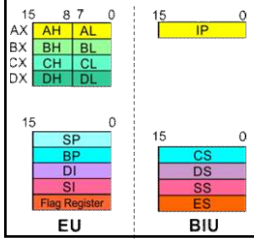
- **16-bit**
- **Points to the current stack.**
- **The 20-bit physical stack address is calculated from the Stack Segment (SS) and the Stack Pointer (SP) for stack instructions such as PUSH and POP.**
- **In based addressing mode, the 20-bit physical stack address is calculated from the Stack segment (SS) and the Base Pointer (BP).**



Segment Registers

Extra Segment Register

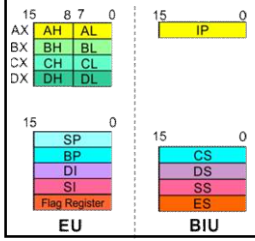
- **16-bit**
- **Points to the extra segment in which data (in excess of 64K pointed to by the DS) is stored.**
- **String instructions use the ES and DI to determine the 20-bit physical address for the destination.**



Segment Registers

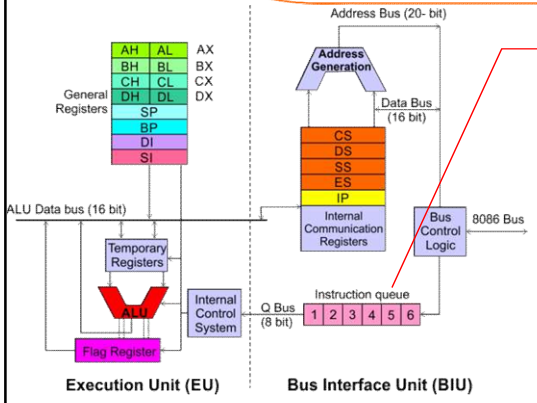
Instruction Pointer

- **16-bit**
- **Always points to the next instruction to be executed within the currently executing code segment.**
- **So, this register contains the 16-bit offset address pointing to the next instruction code within the 64Kb of the code segment area.**
- **Its content is automatically incremented as the execution of the next instruction takes place.**



8086 Microprocessor

Architecture — Bus Interface Unit (BIU)



Instruction queue

- A group of First-In-First-Out (FIFO) in which up to 6 bytes of instruction code are pre fetched from the memory ahead of time.
- This is done in order to speed up the execution by overlapping instruction fetch with execution.
- This mechanism is known as pipelining.

28

8086 Microprocessor

Architecture — Execution Unit (EU)

EU decodes and executes instructions.

A decoder in the EU control system translates instructions.

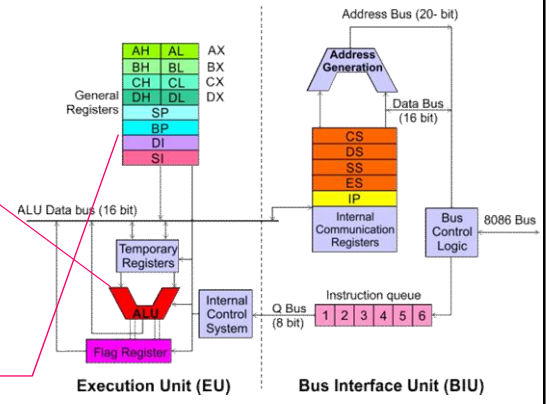
16-bit ALU for performing arithmetic and logic operation

Four general purpose registers(AX, BX, CX, DX);

Pointer registers (Stack Pointer, Base Pointer);

and

Index registers (Source Index, Destination Index) each of 16-bits



Some of the 16 bit registers can be used as two 8 bit registers as :

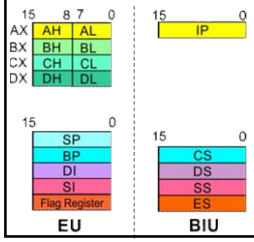
AX can be used as AH and AL
BX can be used as BH and BL
CX can be used as CH and CL
DX can be used as DH and DL

29

EU Registers

Accumulator Register (AX)

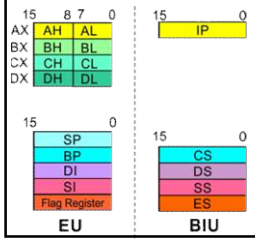
- Consists of two 8-bit registers AL and AH, which can be combined together and used as a 16-bit register AX.
- AL in this case contains the low order byte of the word, and AH contains the high-order byte.
- The I/O instructions use the AX or AL for inputting / outputting 16 or 8 bit data to or from an I/O port.
- Multiplication and Division instructions also use the AX or AL.



EU Registers

Base Register (BX)

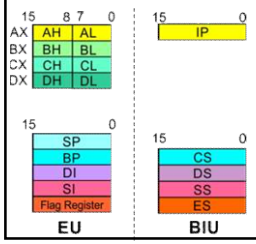
- Consists of two 8-bit registers BL and BH, which can be combined together and used as a 16-bit register BX.
- BL in this case contains the low-order byte of the word, and BH contains the high-order byte.
- This is the only general purpose register whose contents can be used for addressing the 8086 memory.
- All memory references utilizing this register content for addressing use DS as the default segment register.



EU Registers

Counter Register (CX)

- 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.
- Instructions such as **SHIFT**, **ROTATE** and **LOOP** use the contents of CX as a counter.



Example:

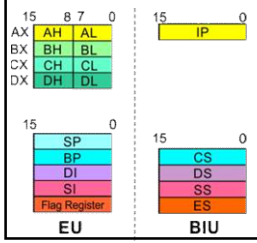
The instruction **LOOP START** automatically decrements CX by 1 without affecting flags and will check if [CX] = 0.

If it is zero, 8086 executes the next instruction; otherwise the 8086 branches to the label START.

32

EU Registers

Data Register (DX)

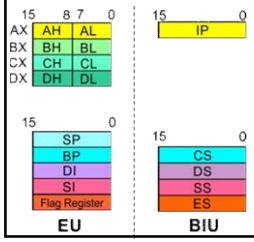


33

EU Registers

Stack Pointer (SP) and Base Pointer (BP)

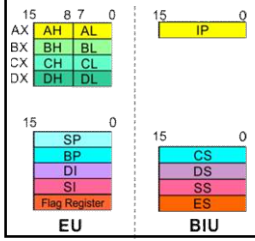
- SP and BP are used to access data in the stack segment.
- SP is used as an offset from the current SS during execution of instructions that involve the stack segment in the external memory.
- SP contents are automatically updated (incremented/ decremented) due to execution of a POP or PUSH instruction.
- BP contains an offset address in the current SS, which is used by instructions utilizing the based addressing mode.



EU Registers

Source Index (SI) and Destination Index (DI)

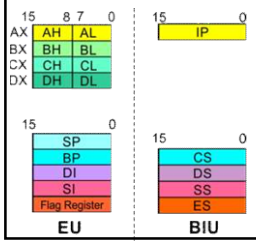
- Used in indexed addressing.
- Instructions that process data strings use the SI and DI registers together with DS and ES respectively in order to distinguish between the source and destination addresses.



EU Registers

Source Index (SI) and Destination Index (DI)

- **Used in indexed addressing.**
- **Instructions that process data strings use the SI and DI registers together with DS and ES respectively in order to distinguish between the source and destination addresses.**



Flag Register

Auxiliary Carry Flag

This is set, if there is a carry from the lowest nibble, i.e, bit three during addition, or borrow for the lowest nibble, i.e, bit three, during subtraction.

Carry Flag

This flag is set, when there is a carry out of MSB in case of addition or a borrow in case of subtraction.

Sign Flag

This flag is set, when the result of any computation is negative

Zero Flag

This flag is set, if the result of the computation or comparison performed by an instruction is zero

Parity Flag

This flag is set to 1, if the lower byte of the result contains even number of 1's ; for odd number of 1's set to zero.

Over flow Flag

This flag is set, if an overflow occurs, i.e, if the result of a signed operation is large enough to accommodate in a destination register. The result is of more than 7-bits in size in case of 8-bit signed operation and more than 15-bits in size in case of 16-bit sign operations, then the overflow will be set.

Tarp Flag

If this flag is set, the processor enters the single step execution mode by generating internal interrupts after the execution of each instruction

Direction Flag

This is used by string manipulation instructions. If this flag bit is '0', the string is processed beginning from the lowest address to the highest address, i.e., auto incrementing mode. Otherwise, the string is processed from the highest address towards the lowest address, i.e., auto incrementing mode.

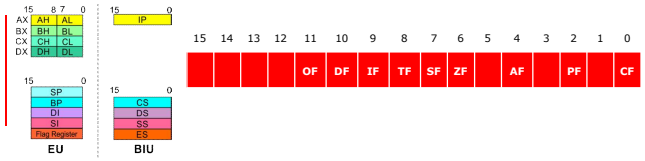
Interrupt Flag

Causes the 8086 to recognize external mask interrupts; clearing IF disables these interrupts.

8086 Microprocessor

Architecture

8086 registers categorized into 4 groups



Sl.No.	Type	Register width	Name of register
1	General purpose register	16 bit	AX, BX, CX, DX
		8 bit	AL, AH, BL, BH, CL, CH, DL, DH
2	Pointer register	16 bit	SP, BP
3	Index register	16 bit	SI, DI
4	Instruction Pointer	16 bit	IP
5	Segment register	16 bit	CS, DS, SS, ES
6	Flag (PSW)	16 bit	Flag register

97

8086 Microprocessor

Architecture

Registers and Special Functions

Register	Name of the Register	Special Function
AX	16-bit Accumulator	Stores the 16-bit results of arithmetic and logic operations
AL	8-bit Accumulator	Stores the 8-bit results of arithmetic and logic operations
BX	Base register	Used to hold base value in base addressing mode to access memory data
CX	Count Register	Used to hold the count value in SHIFT, ROTATE and LOOP instructions
DX	Data Register	Used to hold data for multiplication and division operations
SP	Stack Pointer	Used to hold the offset address of top stack memory
BP	Base Pointer	Used to hold the base value in base addressing using SS register to access data from stack memory
SI	Source Index	Used to hold index value of source operand (data) for string instructions
DI	Data Index	Used to hold the index value of destination operand (data) for string operations

98