Subject Name:Microprocessor

Unit No:1 Unit Name: The Intel Microprocessors 8086 Architecture

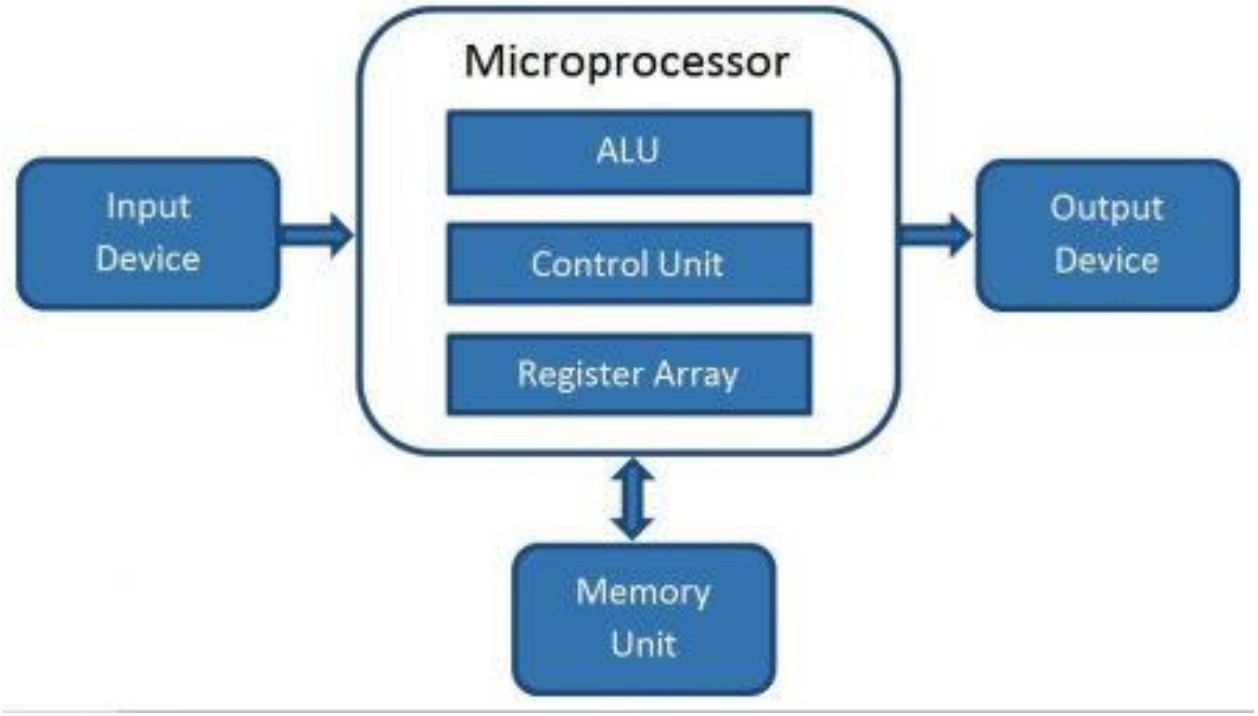
Faculty Name :Dr. Shilpa Shinde
Ms. Ekta Sarda
Mr. Prathmesh Gunjgur

Index - (Heading Font: minion pro, Size:20)

Week No: 1

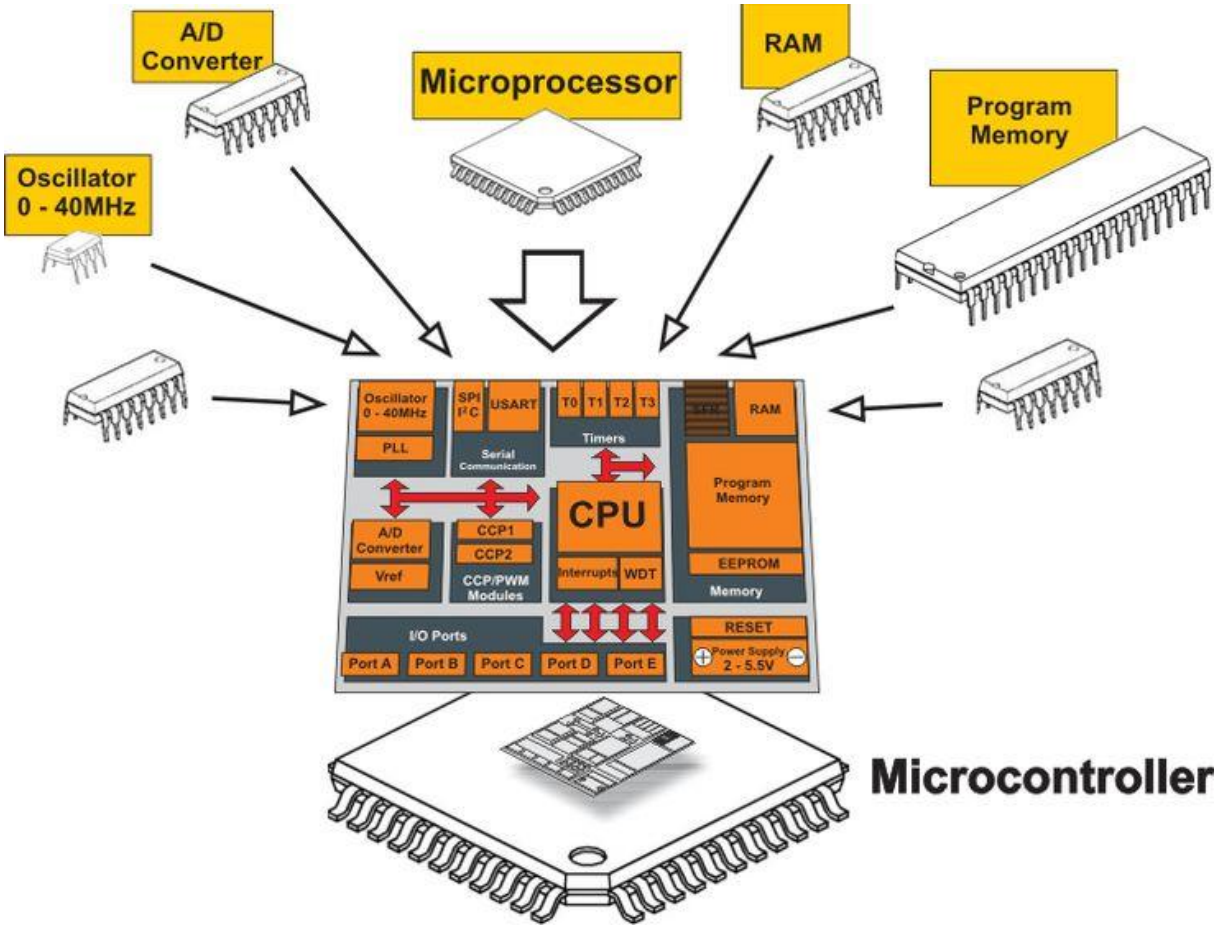
Lecture 1 : 8086 CPU Architecture & Programmer's Model of 8086

Block Diagram of a Computer



Microprocessor Vs. Microcontroller

Microprocessor Vs. Microcontroller



What is Microprocessor ? (NPTEL)

- https://youtu.be/0t4LROuEVnw
- Prof. Shaik Rafi Ahmed – IIT Guwahati

Features of Microprocessor

- **Low Cost** - Due to integrated circuit technology microprocessors are available at very low cost. It will reduce the cost of a computer system.
- **High Speed** - Due to the technology involved in it, the microprocessor can work at very high speed. It can execute millions of instructions per second.
- **Small Size** - A microprocessor is fabricated in a very less footprint due to very large scale and ultra large scale integration technology. Because of this, the size of the computer system is reduced.
- **Versatile** - The same chip can be used for several applications, therefore, microprocessors are versatile.
- **Low Power Consumption** - Microprocessors are using metal oxide semiconductor technology, which consumes less power.
- **Less Heat Generation** - Microprocessors uses semiconductor technology which will not emit much heat as compared to vacuum tube devices.
- **Reliable** - Since microprocessors use semiconductor technology, therefore, the failure rate is very less. Hence it is very reliable.
- **Portable** - Due to the small size and low power consumption microprocessors are portable.

Features of Microprocessor

Low Cost

High Speed

Small Size

Versatile

Low Power Consumption

Reliable

Portable

Educational Need of Microprocessor

- **H/w Designer**.
- **S/w Designer.**
- **System Integrator.**

Applications of Microprocessor

- **General purpose µPs**. – desktop computers, laptops, workstations, servers, super computers
- **Embedded Systems at Home :** A number of modern devices in the home are microprocessor based i.e. camera; washing machines; calculators; hi-fi systems; telephones; microwave ovens; burglar alarms etc. The input are usually simple numeric keyboards, sensors, buttons or while the output include lights, simple LCD screens displays, motors and relays, LEDs, buzzers etc.
- **Industrial Applications of Microprocessors :** Some industrial items which use microprocessors technology include: cars, boats, planes, trucks, heavy machinery, elevators, gasoline pumps, credit-card processing units, traffic control devices, computer servers, most high tech medical devices, surveillance systems, security systems, and even some doors with automatic entry.
- **Transportation Industry :** Automobiles, trains and planes also use microprocessor technology. Consumer vehicles-buses, cars, trucks -integrate microprocessors to communicate important information throughout the vehicle. E.g., navigation systems provide information using microprocessors and global positioning system (GPS) technology.

Applications of Microprocessor

- **In Medicals :** Many medical devices, like an insulin pump, are typically controlled by a microprocessor. The microprocessors perform various functions, such as processing data from bio-sensors, storing measurements, and analyzing results.
- **Instrumentation :** Microprocessor is also very useful in the field of instrumentation. Function generators, frequency counters, frequency synthesizers, spectrum analyses and many other instruments are available, when microprocessors are used as controller.
- **Office Automation and Publication :** Microprocessor based system with software packages has changed the office environment. Microprocessors based systems are being used for spread sheet operations, word processing, storage etc. The Publication technology has revolutionized by the microprocessor.
- **Communication :** In communication the telephone industry is most important. In this industry, microprocessors are used in digital telephone sets, telephone exchanges and modem etc. The use of microprocessor in satellite communication, television, has made teleconferencing possible. Railway reservation and airline reservation system also uses microprocessor technology. WAN (Wide Area Network) and LAN (Local Area Network) for communication of vertical information through computer network.

Applications of Microprocessor

8086 Microprocessor

- Intel 8086 microprocessor is the enhanced version of Intel 8085 microprocessor. It was designed by Intel in 1976.
- It is a **16-bit** microprocessor.
- Intel 8086 is built on a single semiconductor chip and packaged in a 40-pin IC package. The type of package is DIP (Dual Inline Package).
- Intel 8086 uses 20 address lines and 16 data- lines. It can directly address up to 2^{20} = 1 Mbyte of memory.
- 8086 is designed to operate in two modes, i.e., Minimum and Maximum mode.
- It can support up to 64K I/O ports.
- It provides 14, 16 -bit registers.

Features of 8086 Microprocessor

- It has an instruction queue, which is capable of storing six instruction bytes from the memory resulting in faster processing.
- It was the first 16-bit processor having 16-bit ALU, 16-bit registers, internal data bus, and 16-bit external data bus resulting in faster processing.
- It is available in 3 versions based on the frequency of operation −
  - 8086 → 5MHz
  - 8086-2 → 8MHz
  - 8086-1 → 10 MHz

Fact : June 5, 2018, Intel released a limited-edition CPU celebrating the 40th anniversary of the Intel 8086, called the Intel Core i7-8086K.

- It uses two stages of pipelining, i.e. Fetch Stage and Execute Stage, which improves performance.

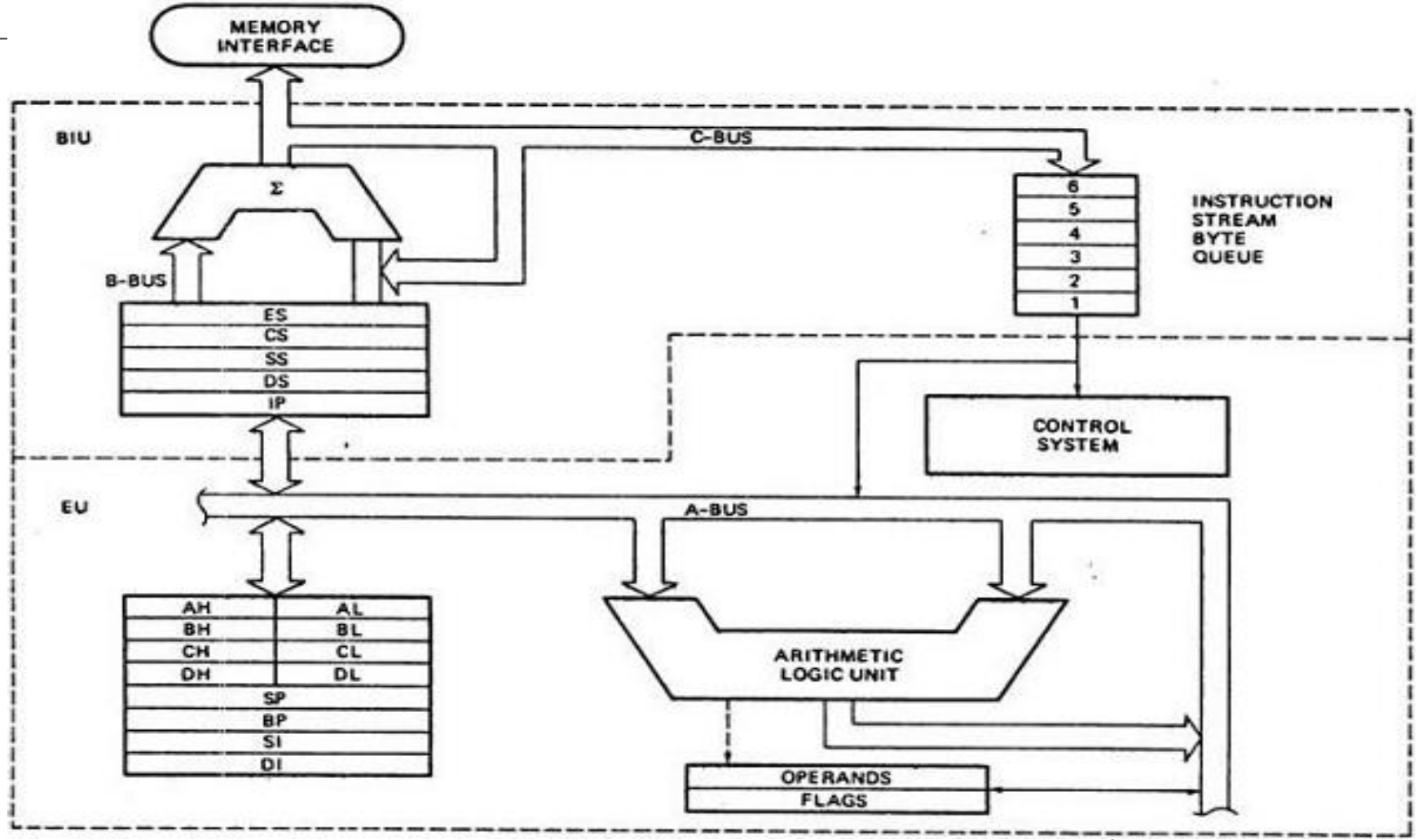
Features of 8086 Microprocessor

- Fetch stage can prefetch up to 6 bytes of instructions and stores them in the queue.
- It has multiplexed address and data bus AD_0- AD_{15} and A_{16} – A_{19}.
- It requires single phase clock with **33% duty cycle** to provide internal timing.
- Execute stage executes these instructions.
- It has 256 vectored interrupts.
- It consists of 29,000 transistors.

CPU Fetch-Decode-Execute

- https://www.hartismere.com/20398/CPU-Fetch-Decode-Execute-Animation

Internal Architecture of 8086



Internal Architecture of 8086

- 8086 Microprocessor is divided into two functional units, i.e.,
  - EU (Execution Unit)
  - BIU (Bus Interface Unit).

A. EU (Execution Unit)

Execution unit gives instructions to BIU stating from where to fetch the data and then decode and execute those instructions. Its function is to control operations on data using the instruction decoder & ALU. EU has no direct connection with system buses as shown in the figure, it performs operations over data through BIU.

- Let us now discuss the functional parts of 8086 microprocessors.

Internal Architecture of 8086

- **ALU**
  It handles all arithmetic and logical operations, like +, −, ×, /, OR, AND, NOT operations.

- **Flag Register**
  It is a 16-bit register that behaves like a flip-flop, i.e. it changes its status according to the result stored in the accumulator. It has 9 flags and they are divided into 2 groups − Conditional Flags and Control Flags.

- **Conditional Flags**
  It represents the result of the last arithmetic or logical instruction executed. Following is the list of conditional flags −

  - Carry flag − This flag indicates an overflow condition for arithmetic operations.

Internal Architecture of 8086

- Auxiliary flag – When an operation is performed at ALU, it results in a carry/barrow from lower nibble (i.e. D0 – D3) to upper nibble (i.e. D4 – D7), then this flag is set, i.e. carry given by D3 bit to D4 is AF flag. The processor uses this flag to perform binary to BCD conversion.

- Parity flag – This flag is used to indicate the parity of the result, i.e. when the lower order 8-bits of the result contains even number of 1's, then the Parity Flag is set. For odd number of 1's, the Parity Flag is reset.

- Zero flag – This flag is set to 1 when the result of arithmetic or logical operation is zero else it is set to 0.

- Sign flag – This flag holds the sign of the result, i.e. when the result of the operation is negative, then the sign flag is set to 1 else set to 0.

- Overflow flag – This flag represents the result when the system capacity is exceeded.

- **Control Flags**

  Control flags controls the operations of the execution unit. Following is the list of control flags –

  - Trap flag − It is used for single step control and allows the user to execute one instruction at a time for debugging. If it is set, then the program can be run in a single step mode.

  - Interrupt flag − It is an interrupt enable/disable flag, i.e. used to allow/prohibit the interruption of a program. It is set to 1 for interrupt enabled condition and set to 0 for interrupt disabled condition.

  - Direction flag − It is used in string operation. As the name suggests when it is set then string bytes are accessed from the higher memory address to the lower memory address and vice-a-versa.

Internal Architecture of 8086

- **General purpose register**
  There are 8 general purpose registers, i.e., AH, AL, BH, BL, CH, CL, DH, and DL. These registers can be used individually to store 8-bit data and can be used in pairs to store 16bit data. The valid register pairs are AH and AL, BH and BL, CH and CL, and DH and DL. It is referred to the AX, BX, CX, and DX respectively.
  - AX register − It is also known as accumulator register. It is used to store operands for arithmetic operations.
  - BX register − It is used as a base register. It is used to store the starting base address of the memory area within the data segment.
  - CX register − It is referred to as counter. It is used in loop instruction to store the loop counter.
  - DX register − This register is used to hold I/O port address for I/O instruction.

- **Stack pointer register**
  It is a 16-bit register, which holds the address from the start of the segment to the memory location, where a word was most recently stored on the stack.

B. BIU (Bus Interface Unit)

- BIU takes care of all data and addresses transfers on the buses for the EU like sending addresses, fetching instructions from the memory, reading data from the ports and the memory as well as writing data to the ports and the memory. EU has no direction connection with System Buses so this is possible with the BIU. EU and BIU are connected with the Internal Bus.

Internal Architecture of 8086

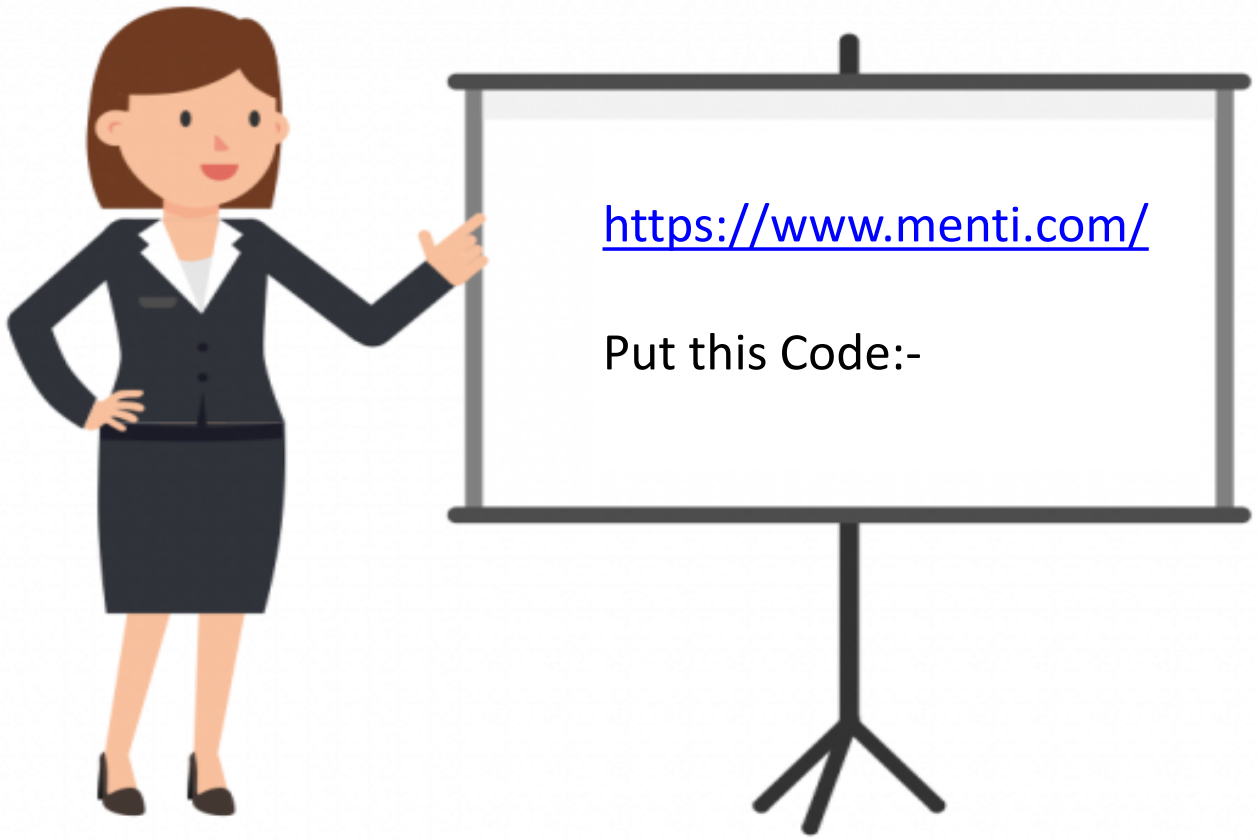
- It has the following functional parts −
  - **Instruction queue** − BIU contains the instruction queue. BIU gets upto 6 bytes of next instructions and stores them in the instruction queue. When EU executes instructions and is ready for its next instruction, then it simply reads the instruction from this instruction queue resulting in increased execution speed.
  - Fetching the next instruction while the current instruction executes is called pipelining.
  - **Segment register** − BIU has 4 segment buses, i.e. CS, DS, SS & ES. It holds the addresses of instructions and data in memory, which are used by the processor to access memory locations. It also contains 1 pointer register IP, which holds the address of the next instruction to executed by the EU.

Internal Architecture of 8086

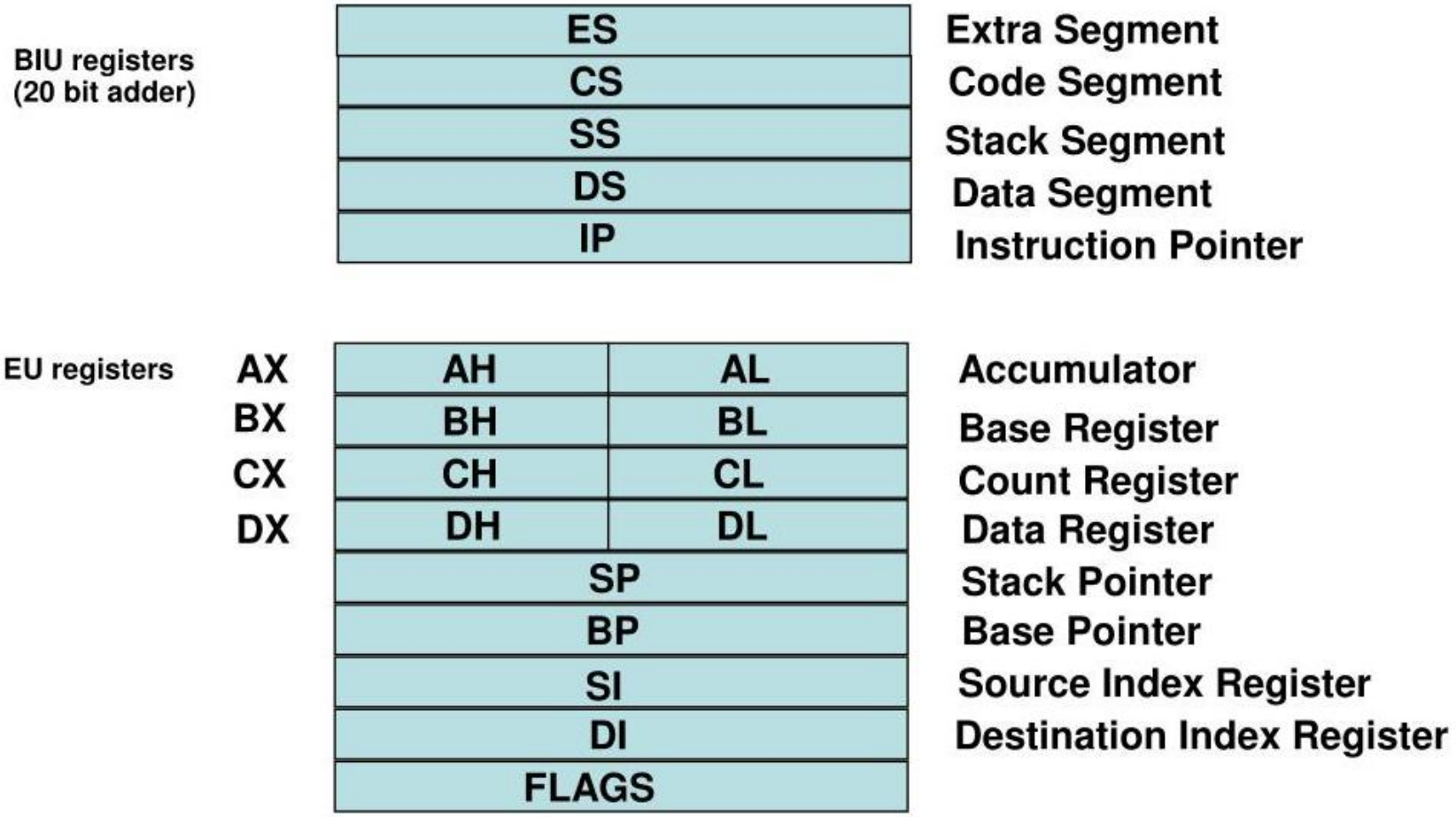
- CS − It stands for Code Segment. It is used for addressing a memory location in the code segment of the memory, where the executable program is stored.
- DS − It stands for Data Segment. It consists of data used by the program and is accessed in the data segment by an offset address or the content of other register that holds the offset address.
- SS − It stands for Stack Segment. It handles memory to store data and addresses during execution.
- ES − It stands for Extra Segment. ES is additional data segment, which is used by the string to hold the extra destination data.

- **Instruction pointer** − It is a 16-bit register used to hold the address of the next instruction to be executed.

Internal Architecture of 8086



Programmer's Model of 8086



Sample Assembly Program in 8086

```
01 DATA SEGMENT
02       NUM1 DB 9H
03       NUM2 DB 7H
04       RESULT DB ?
05 ENDS
06
07 CODE SEGMENT
08     ASSUME DS:DATA CS:CODE
09 START:
10       MOV AX,DATA
11       MOV DS,AX
12
13       MOV AL,NUM1
14       ADD AL,NUM2
15
16       MOV RESULT,AL
17
18       MOV AH,4CH
19       INT 21H
20 ENDS
21 END START
22
```

Segment Registers in 8086

- There are 4 segment registers in 8086 Microprocessor and each of them is of 16 bit. The code and instructions are stored inside these different segments.

1. **Code Segment Register :** It contains all the instructions to be executed. A 16-bit Code Segment register or CS register stores the starting address of the code segment.
2. **Data Segment Register :** It contains data, constants and work areas. A 16-bit Data Segment register or DS register stores the starting address of the data segment.
3. **Stack Segment Register :** It contains data and return addresses of procedures or subroutines. It is implemented as a 'stack' data structure. The Stack Segment register or SS register stores the starting address of the stack.
4. **Extra Segment Register :** It is used by some string operations. The ES register contains the initial address of the extra segment.

General Purpose Registers of 8086

Register	Purpose
AX	Word multiply, word divide, word I /O, Byte multiply, byte divide, byte I/O, decimal arithmetic
AH	Byte multiply, byte divide
BX	Store address information (Indirect Addressing)
CX	String operation, loops
CL	Variable shift and rotate
DX	Word multiply, word divide, indirect I/O (Used to hold I/O address during I/O instructions. If the result is more than 16-bits, the lower order 16-bits are stored in accumulator and higher order 16-bits are stored in DX register)

Pointer & Index Registers in 8086

- used to keep offset addresses.
- Used in various forms of memory addressing.
- In the case of SP and BP the default reference to form a physical address is the Stack Segment (SS-will be discussed under the BIU)
- The index registers (SI & DI) and the BX generally default to the Data segment register (DS).

  - **SP: Stack pointer**
    - Used with SS to access the stack segment

  - **BP: Base Pointer**
    - Primarily used to access data on the stack
    - Can be used to access data in other segments

Pointer & Index Registers in 8086

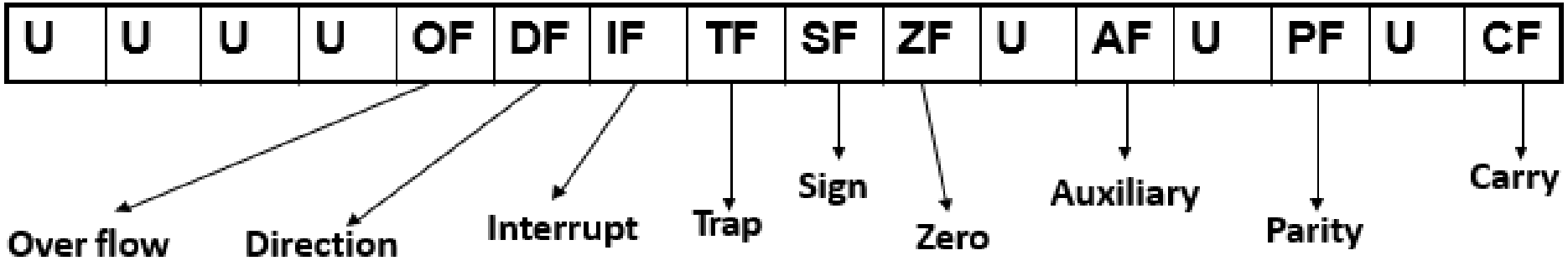
- **SI: Source Index register**
  - is required for some string operations
  - When string operations are performed, the SI register points to memory locations in the data segment which is addressed by the DS register. Thus, SI is associated with the DS in string operations.

- **DI: Destination Index register**
  - is also required for some string operations.
  - When string operations are performed, the DI register points to memory locations in the data segment which is addressed by the ES register. Thus, DI is associated with the ES in string operations.

- The SI and the DI registers may also be used to access data stored in arrays

Flag Registers in 8086

- A flag is a flip flop which indicates some conditions produced by the execution of an instruction or controls certain operations of the EU .
- In 8086 The EU contains a 16 bit flag register
  - 9 of the 16 are active flags and remaining 7 are undefined.
    - 6 flags indicates some conditions- status flags
    - 3 flags –control Flags



U - Unused

Flag Registers in 8086

Flag	Purpose
Carry (CF)	Holds the carry after addition or the borrow after subtraction. Also indicates some error conditions, as dictated by some programs and procedures .
Parity (PF)	PF=0;odd parity, PF=1;even parity.
Auxiliary (AF)	Holds the carry (half – carry) after addition or borrow after subtraction between bit positions 3 and 4 of the result (for example, in BCD addition or subtraction.)
Zero (ZF)	Shows the result of the arithmetic or logic operation. Z=1; result is zero. Z=0; The result is 0
Sign (SF)	Holds the sign of the result after an arithmetic/logic instruction execution. S=1; negative, S=0; positive

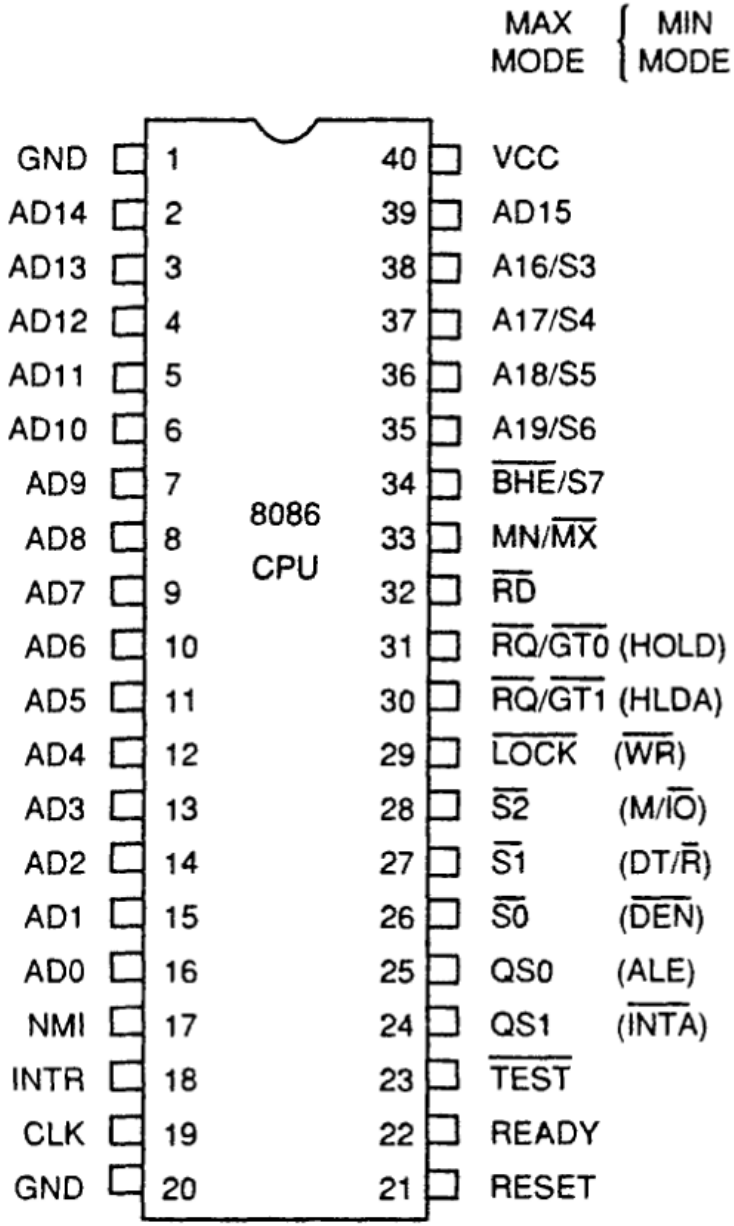
Flag Registers in 8086

Flag	Purpose
Trap (TF)	A control flag. Enables the trapping through an on-chip debugging feature.
Interrupt (IF)	A control flag. Controls the operation of the INTR (interrupt request) I=0; INTR pin disabled. I=1; INTR pin enabled.
Direction (DF)	A control flag. It selects either the increment or decrement mode for DI and /or SI registers during the string instructions.
Overflow (OF)	Overflow occurs when signed numbers are added or subtracted. An overflow indicates the result has exceeded the capacity of the Machine

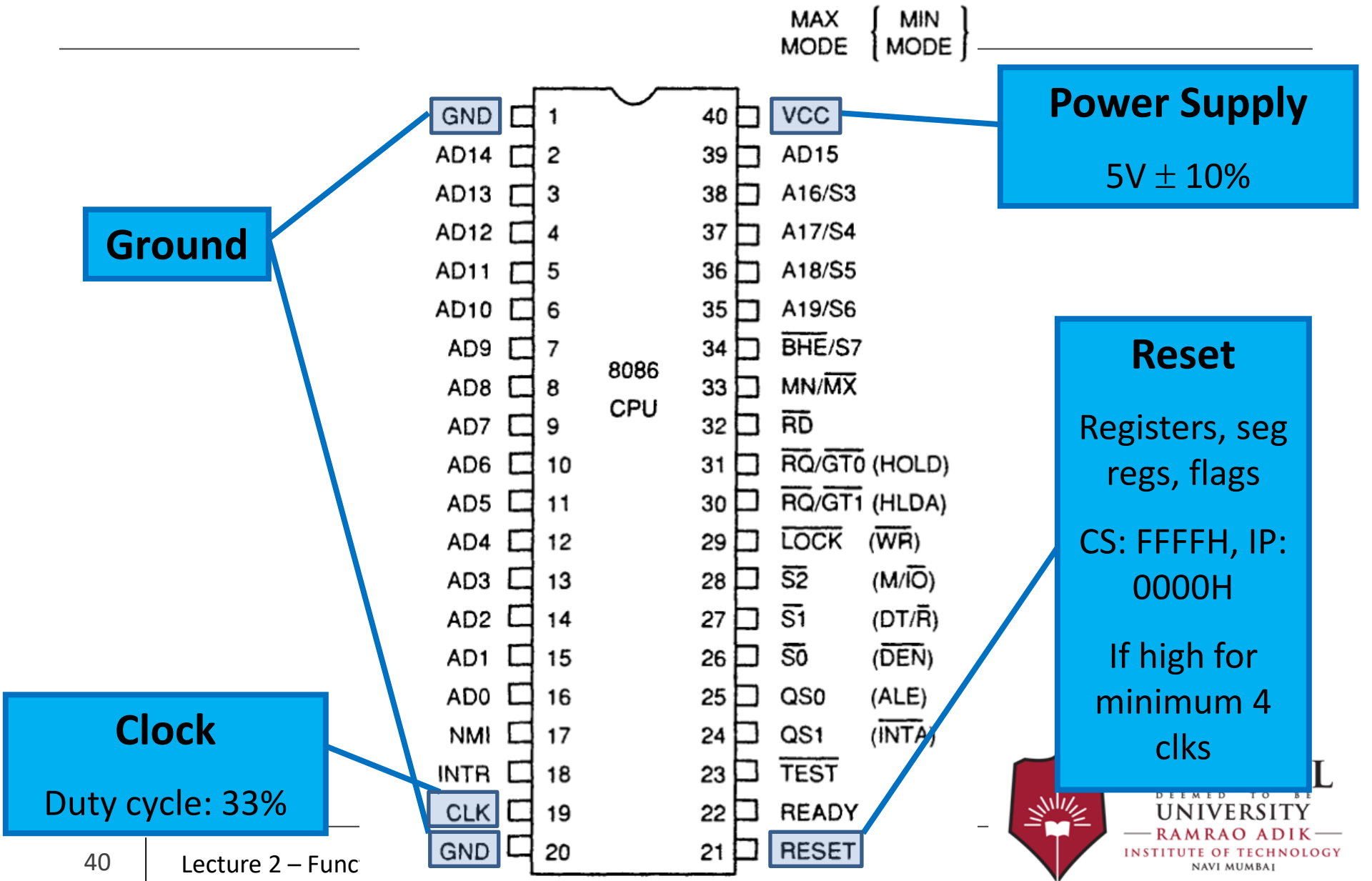
Week No: 1

Lecture 2 : Functional Pin Diagram of 8086

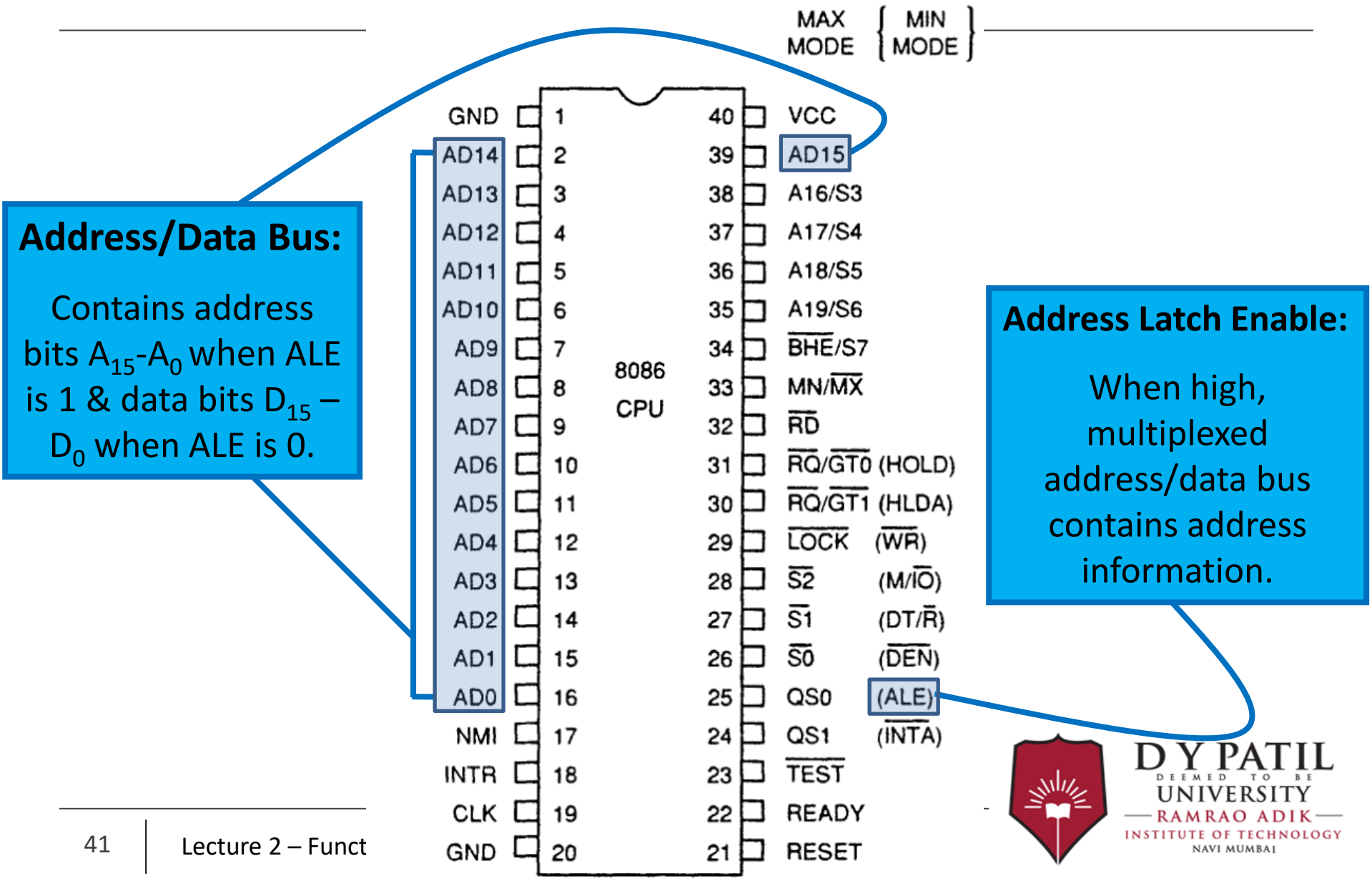
Pin Diagram of 8086



Pin Diagram of 8086



Pin Diagram of 8086



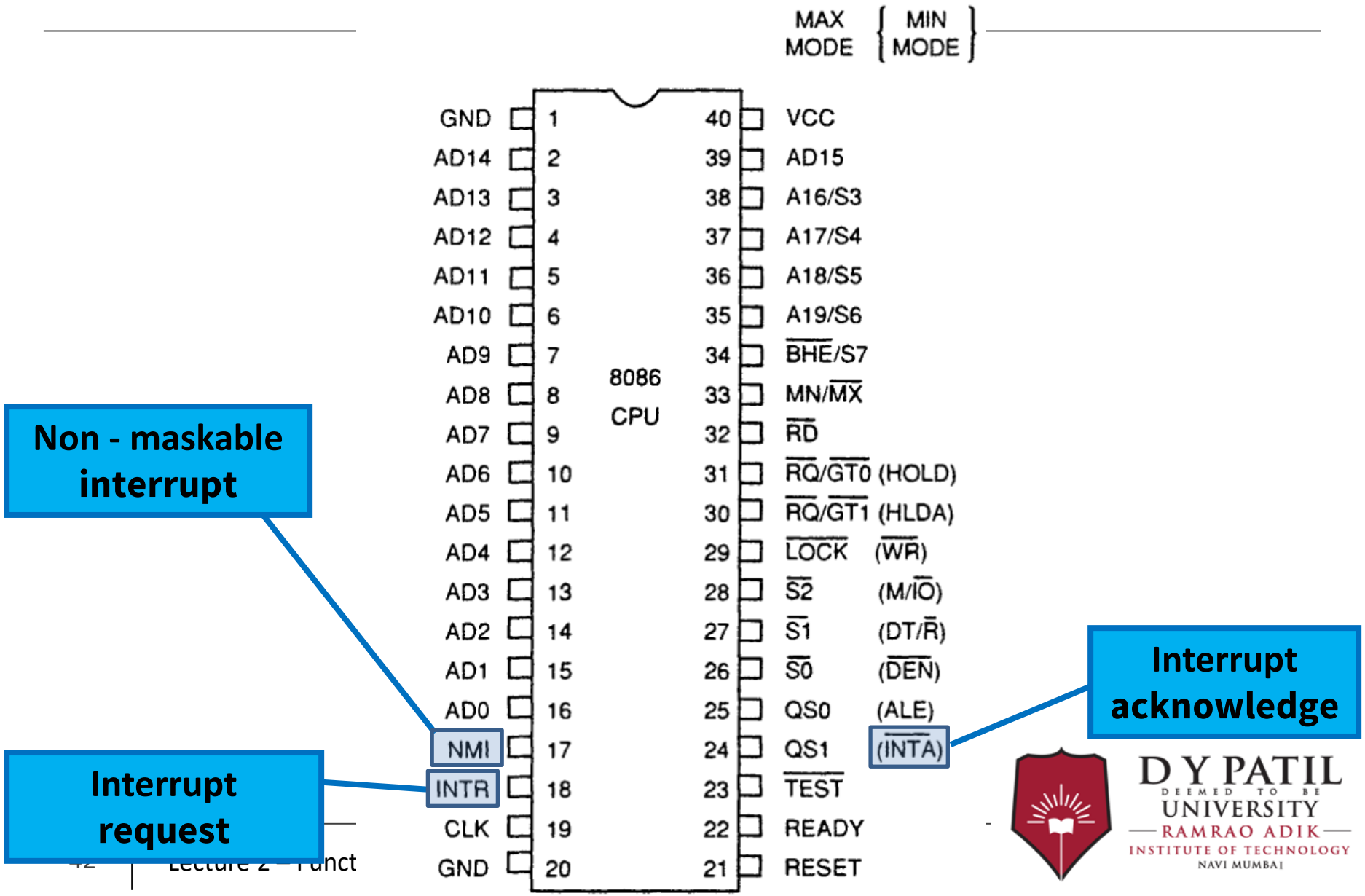
Address/Data Bus:

Contains address bits A_{15}-A_0 when ALE is 1 & data bits D_{15} – D_0 when ALE is 0.

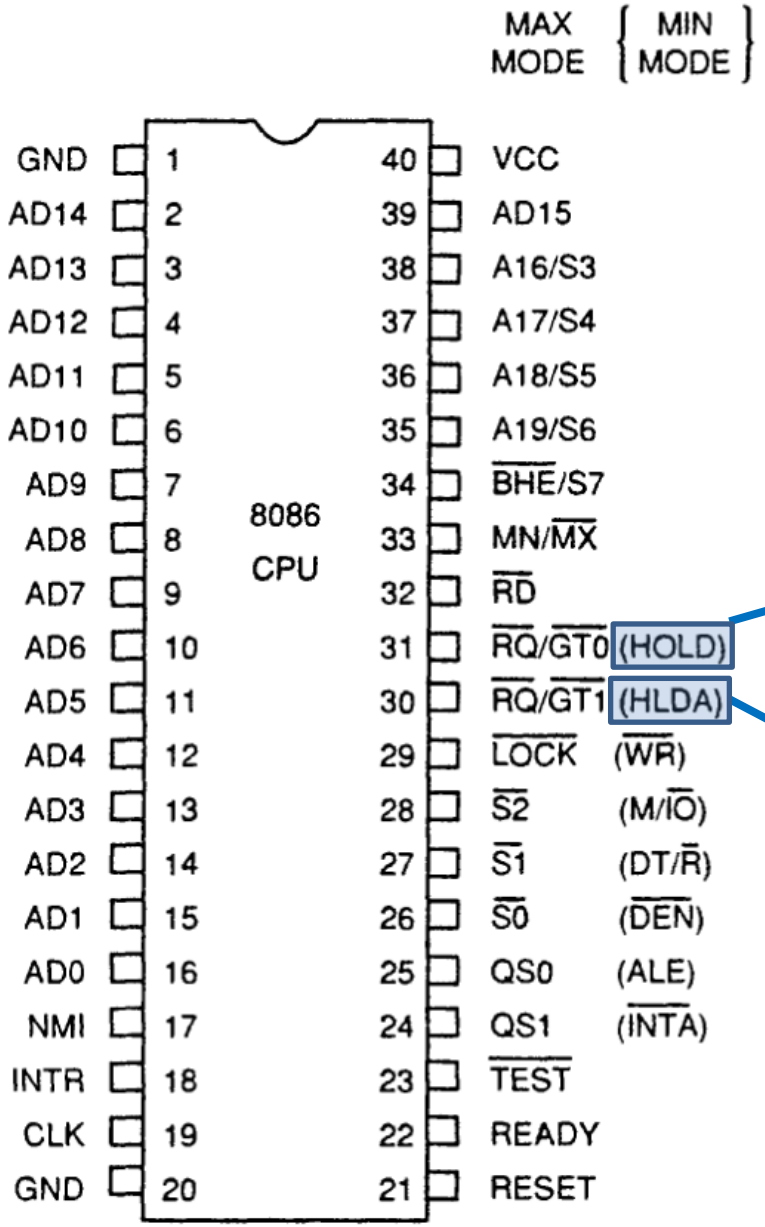
Address Latch Enable:

When high, multiplexed address/data bus contains address information.

Pin Diagram of 8086



Pin Diagram of 8086



Hold

Hold acknowledge

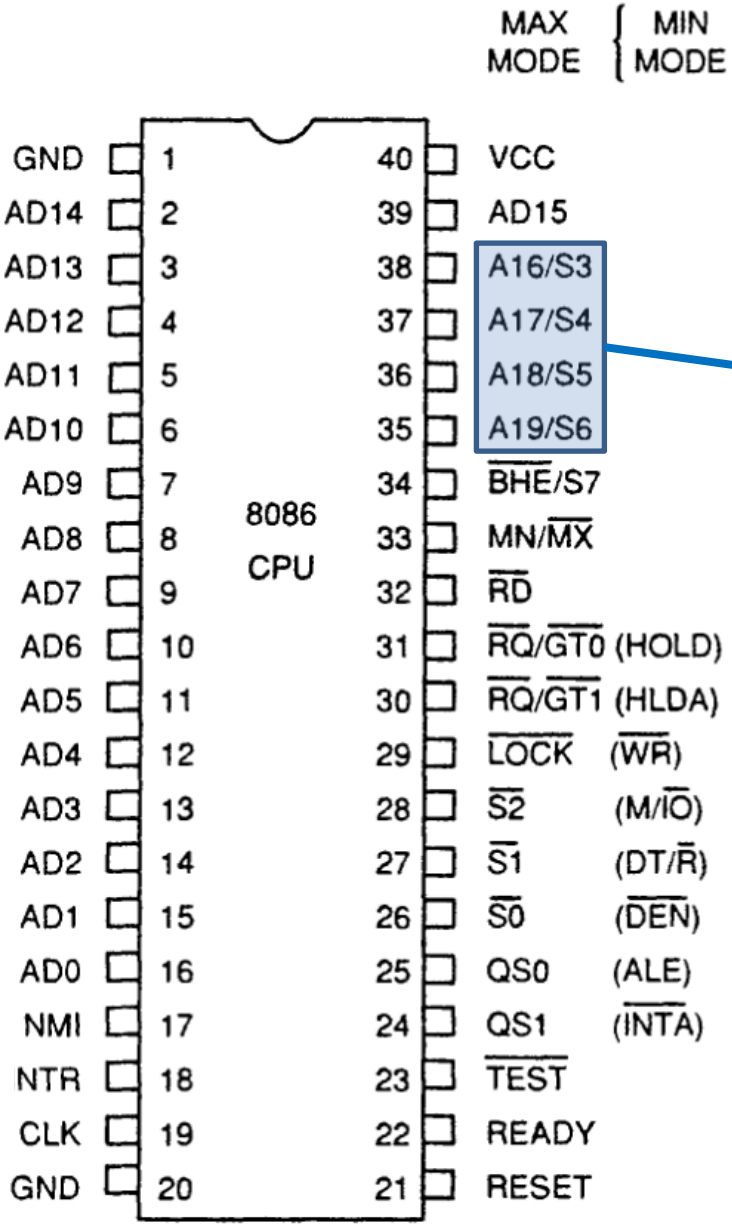
Pin Diagram of 8086

S6: Logic 0.

S5: Indicates condition of IF flag bits.

S4-S3: Indicate which segment is accessed during current bus cycle:

S4	S3	Function
0	0	Extra segment
0	1	Stack segment
1	0	Code or no segment
1	1	Data segment



Address/Status Bus

Address bits A_{19} – A_{16} & Status bits S_6 – S_3

Pin Diagram of 8086

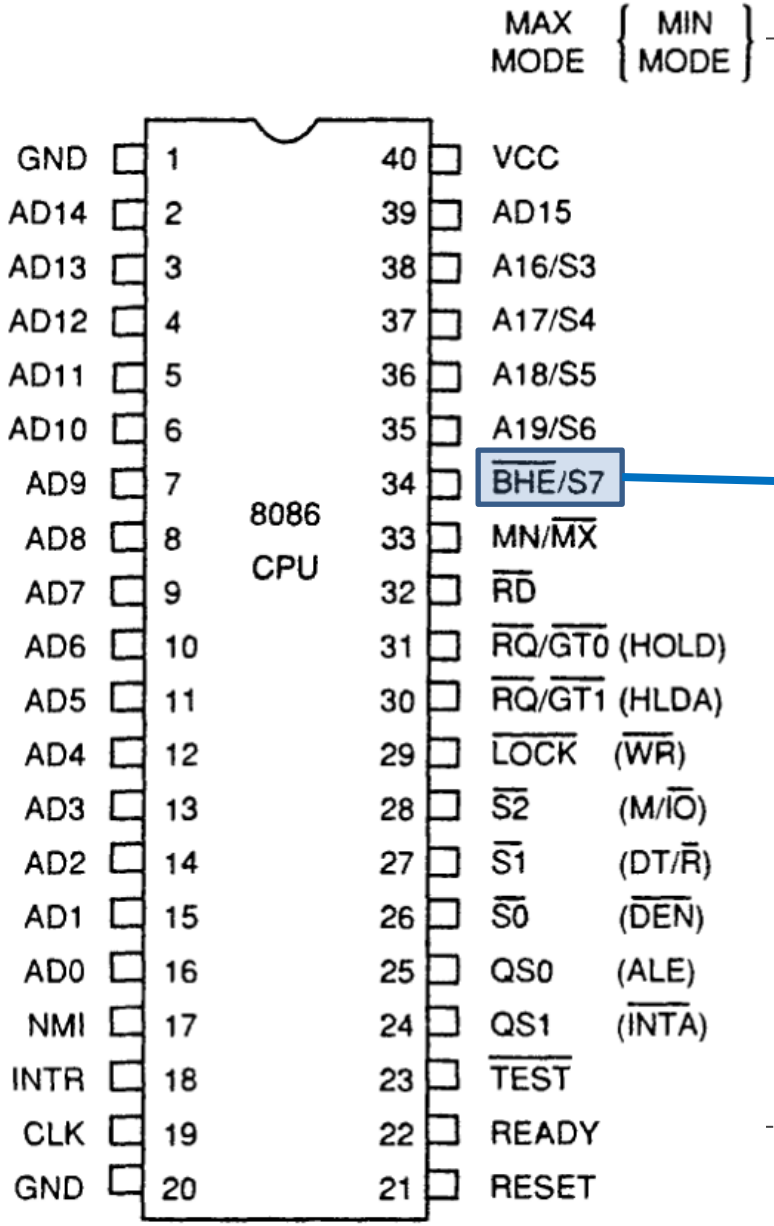
$\overline{BHE}$, A_0:

0,0: Whole word (16-bits)

0,1: High byte to/from odd address

1,0: Low byte to/from even address

1,1: No selection

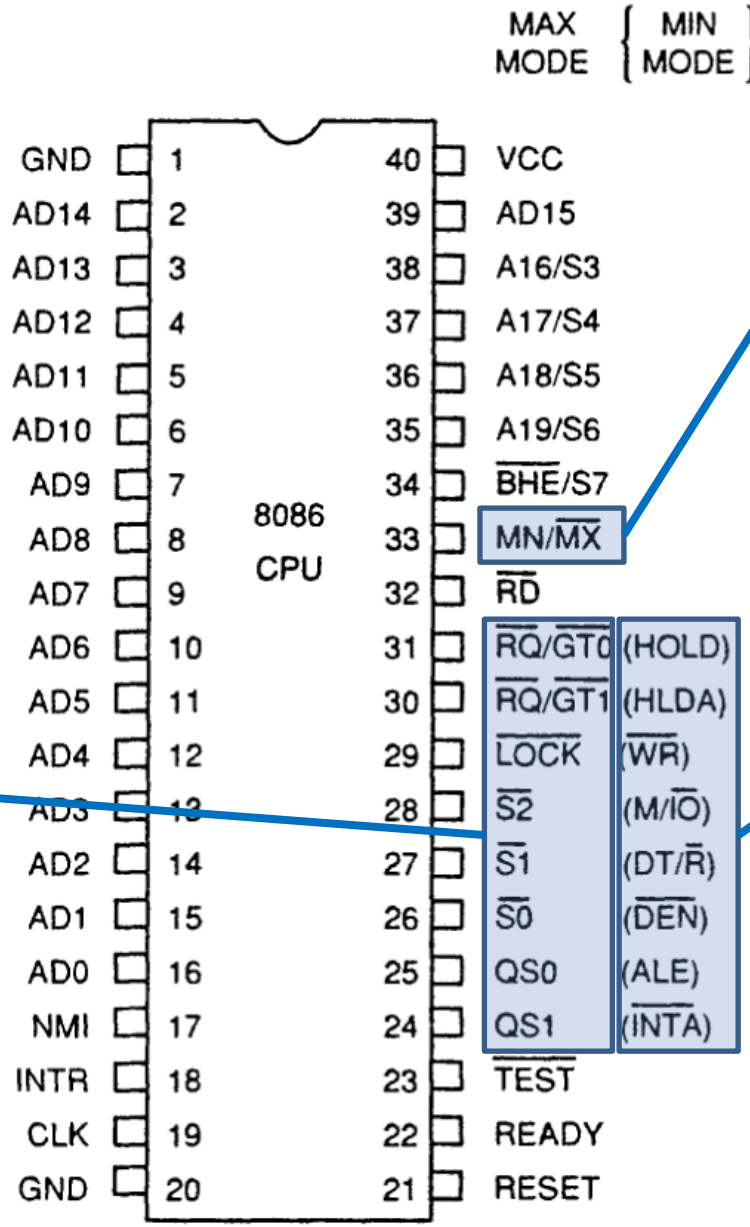


Bus High Enable/S7

Enables most significant data bits $D_{15} – D_8$ during read or write operation.

S_7: Always 1.

Pin Diagram of 8086



Min/Max mode

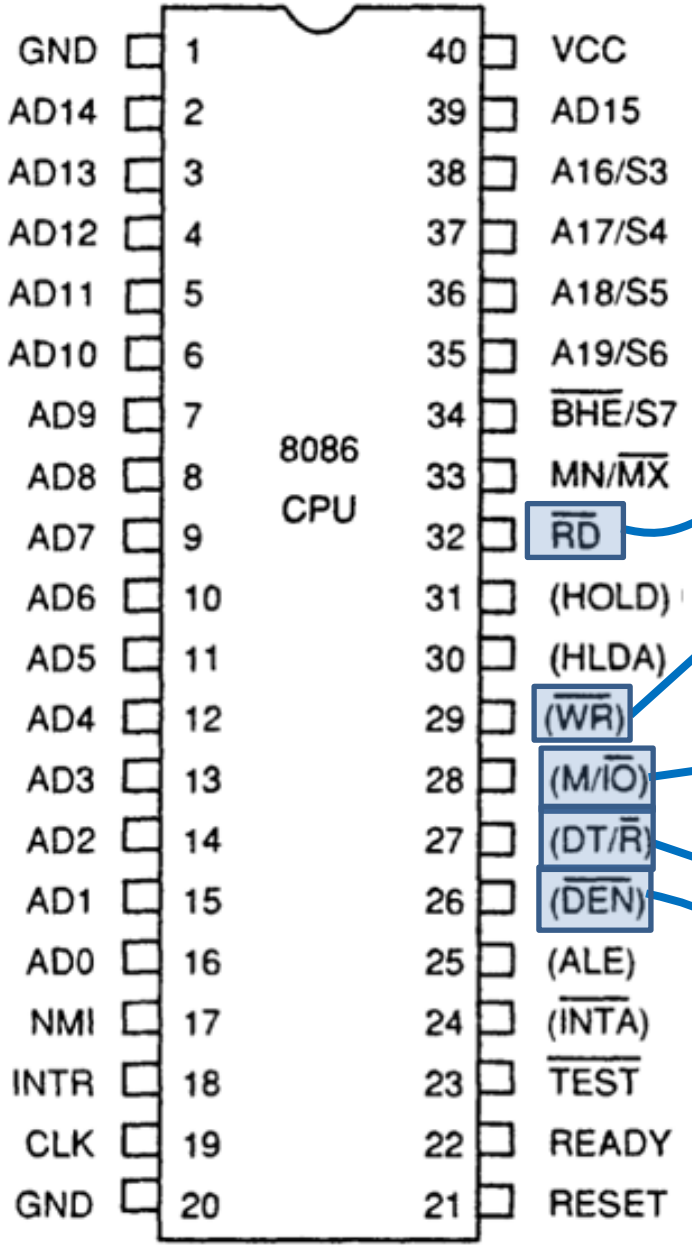
Minimum Mode: +5V

Maximum Mode: 0V

Minimum Mode Pins

Maximum Mode Pins

Minimum Mode Pin Details



Read Signal

Write Signal

Memory or I/0

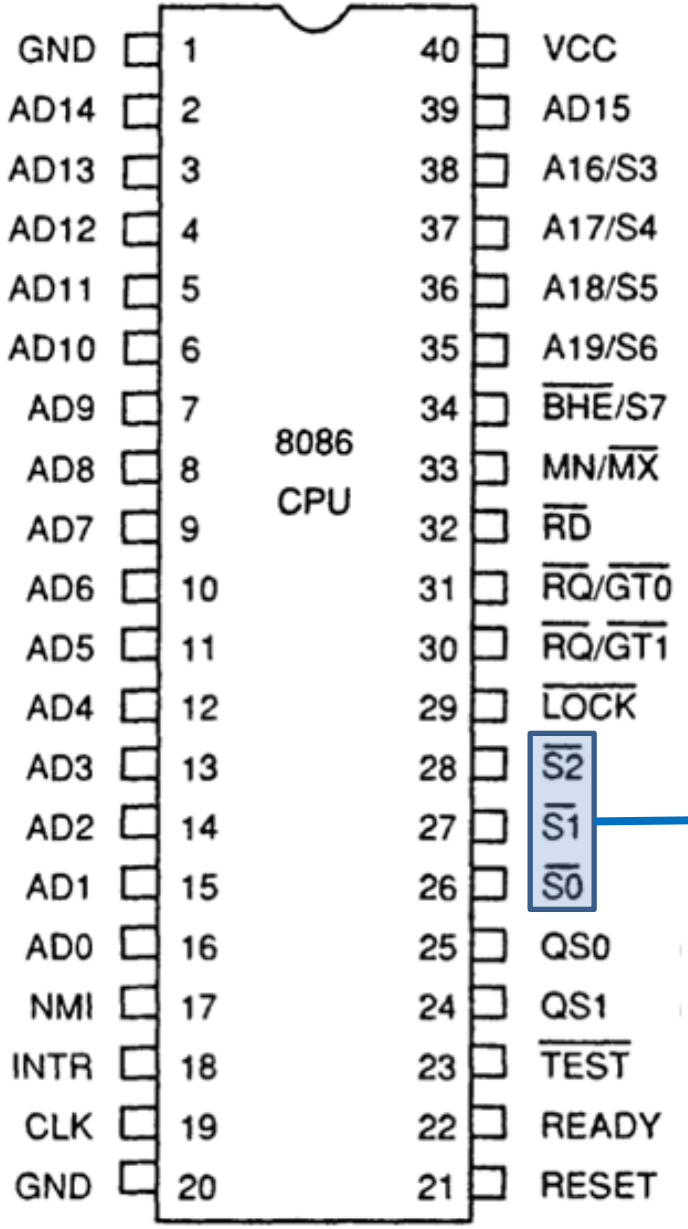
Data Transmit/Receive

Data Bus Enable

Maximum Mode Pin Details

S2 S1 S0

000: INTA
001: read I/O port
010: write I/O port
011: halt
100: code access
101: read memory
110: write memory
111: none -passive



Status Signal

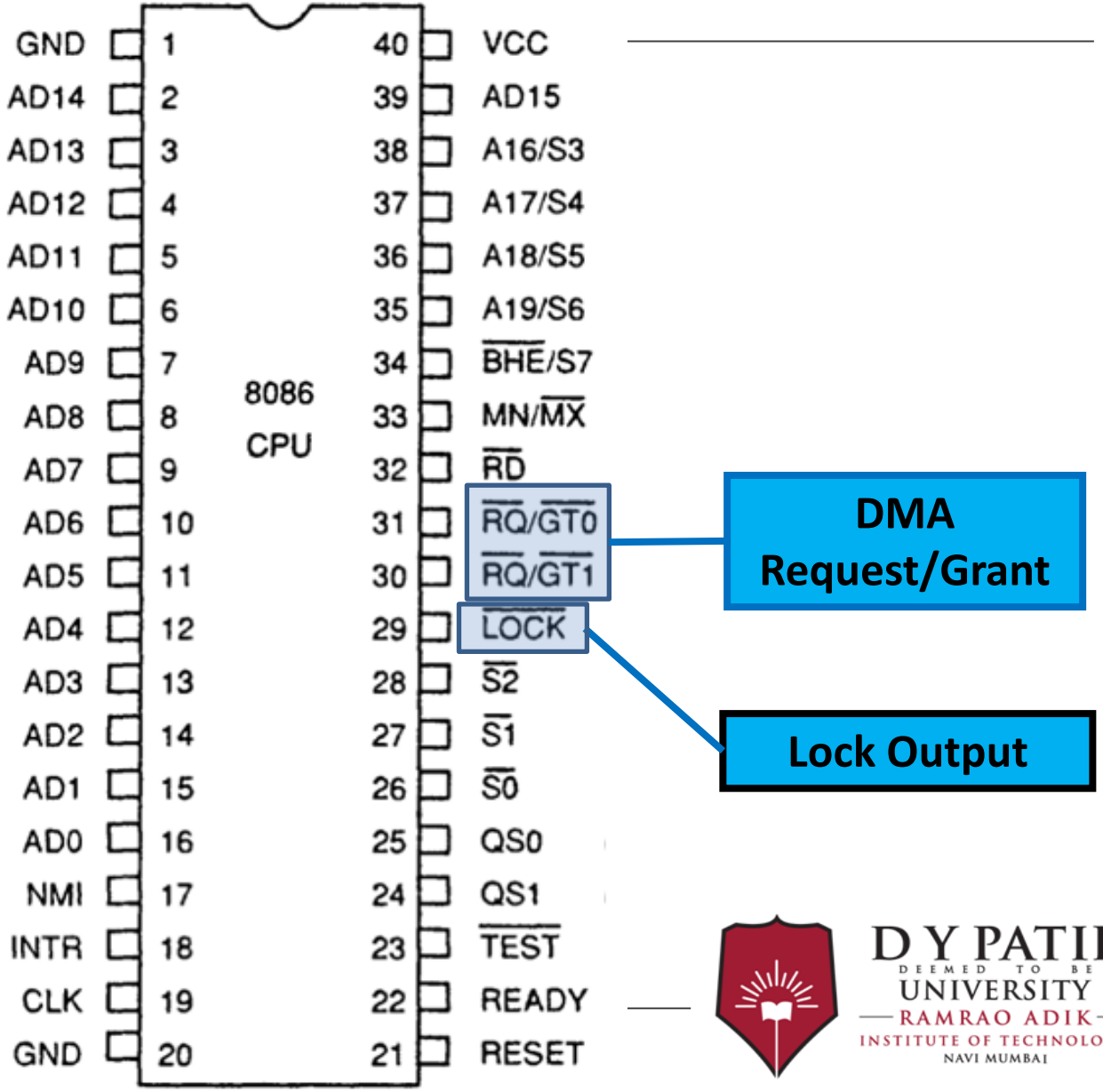
Inputs to 8288 to generate eliminated signals due to max mode.

Maximum Mode Pin Details

Lock Output

Used to lock peripherals off the system

Activated by using the LOCK: prefix on any instruction



Maximum Mode Pin Details

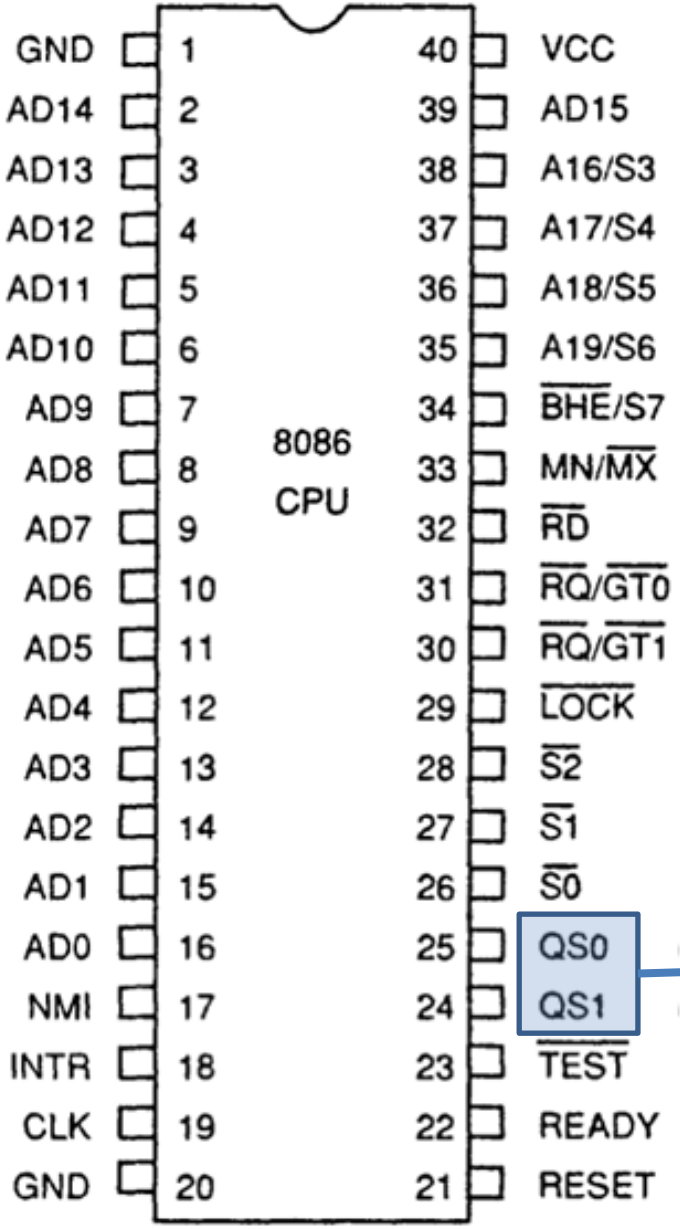
QS1 QS0

00: Queue is idle

01: First byte of opcode

10: Queue is empty

11: Subsequent byte of opcode



Queue Status

Used by numeric coprocessor (8087)

Lets Test What have you Understand Till Now……

- https://puzzel.org/en/jigsaw/play?p=-MB0LpSPl4GZPjRIlnDz
- https://puzzel.org/en/crossword/play?p=-MB0ZhPG65ESy2_YR9AN

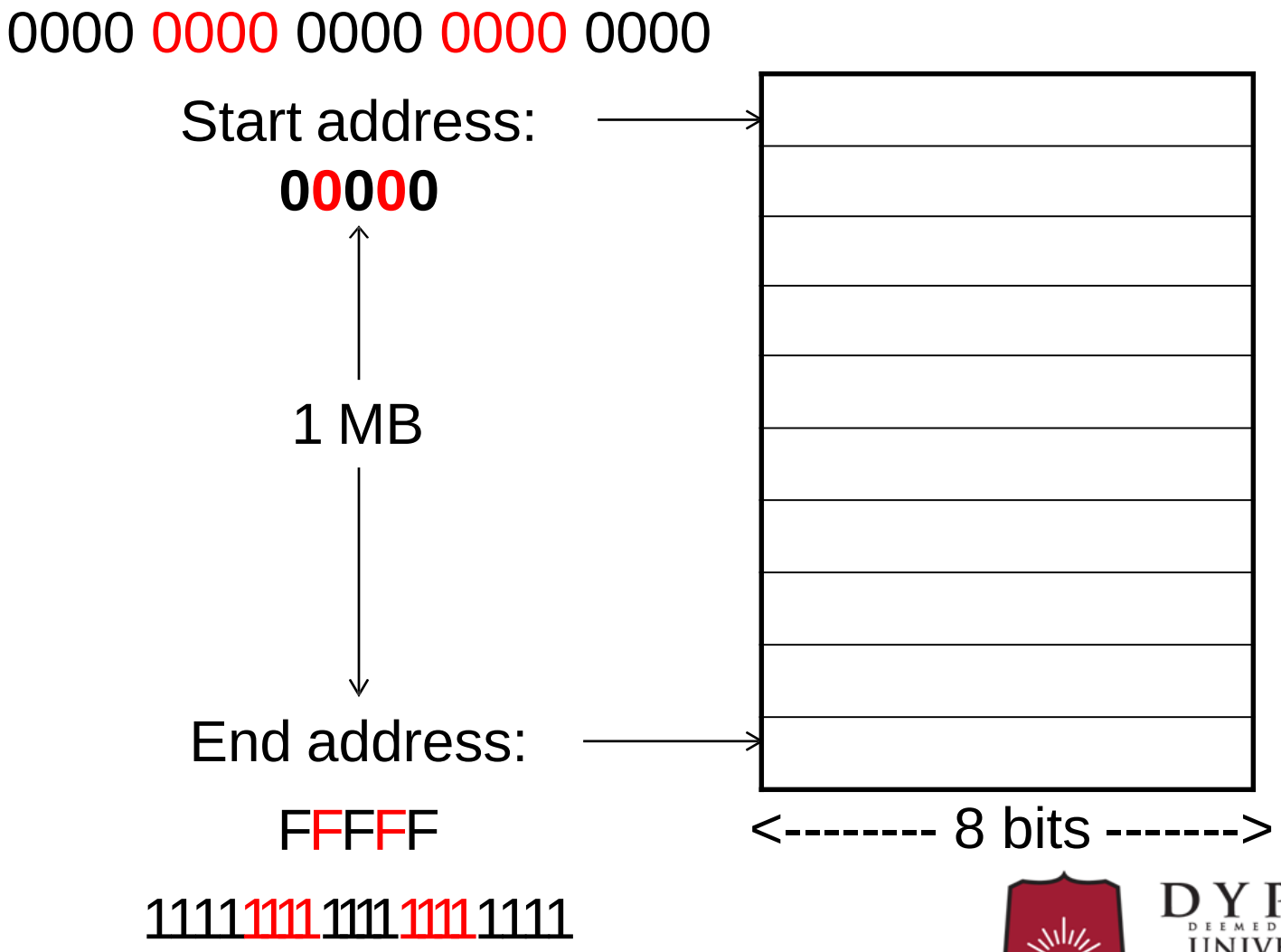
Week No: 1

Lecture 3 : Memory Segmentation

What is Memory ?

- A bank of 1 byte locations, each having its own unique address.
- 2^{20} = 1 MB = 1 Mega Bytes (for 8086 µp)
- Memory lies outside the processor, but it can be accessed by it.
- Memory is used to store programs and data.

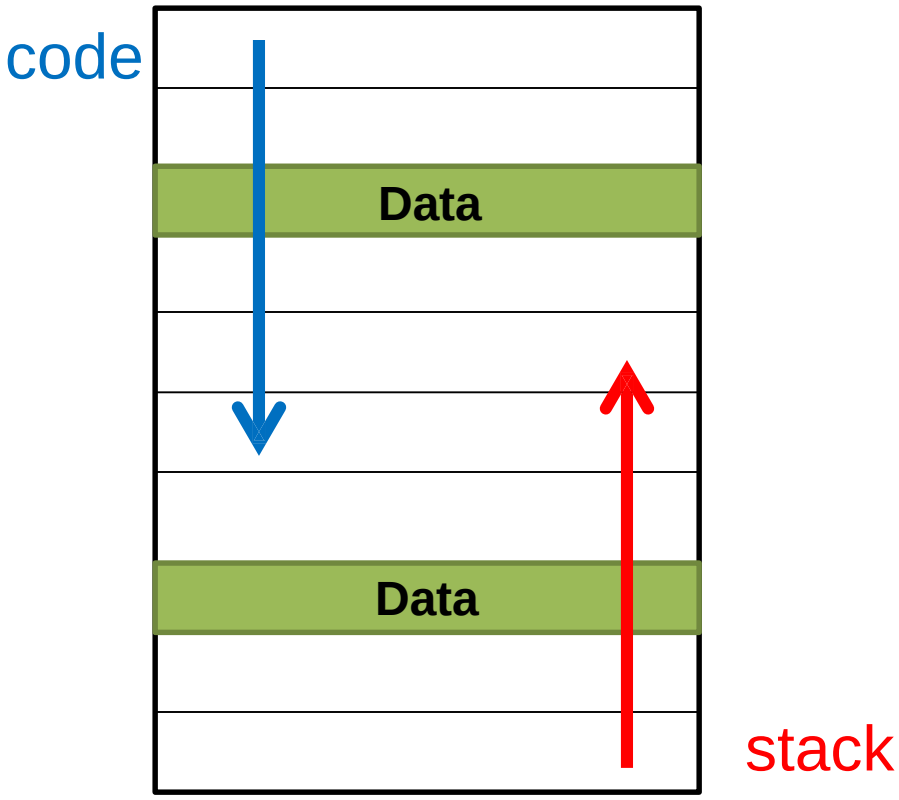
Memory Bank in 8086



How memory is used ?

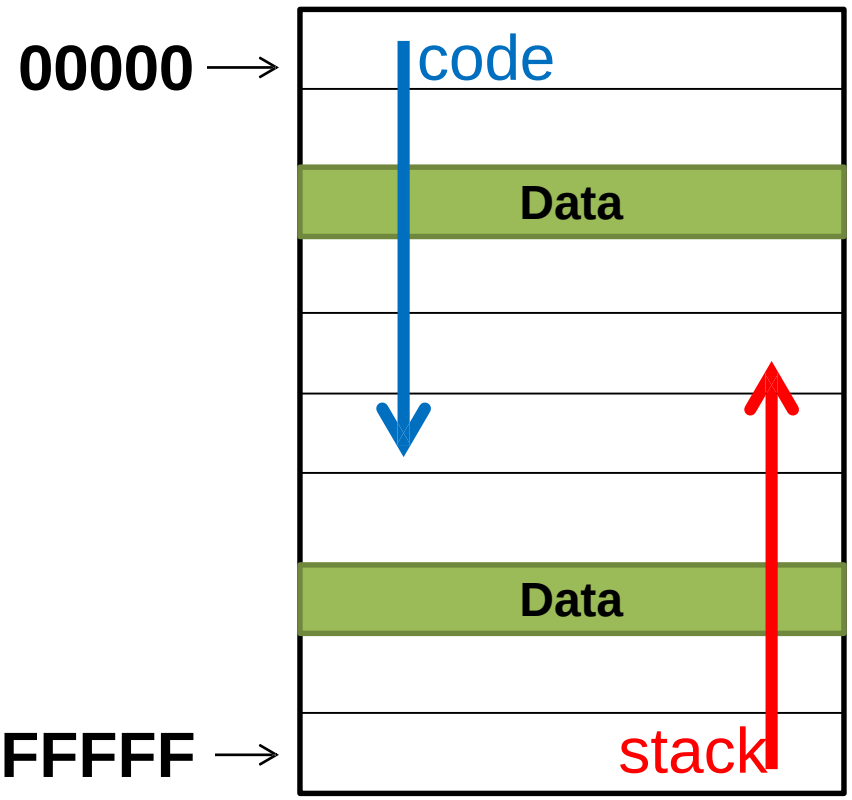
- Programs are stored sequentially from 00000 → FFFFF
- Data can be stored randomly anywhere
- Stack is also stored sequentially, but opposite direction from FFFFF → 0000

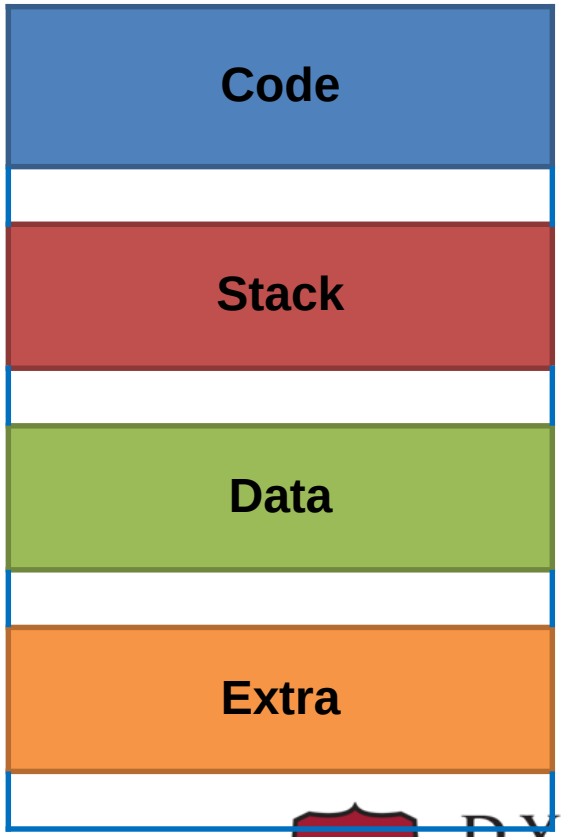
Memory Overwriting



Solution for Memory Overwriting ?

Memory Segmentation





Features

- Prevents over-writing of memory
- Organized management of memory (Gave birth to the concept of files and folders)
- Allows 20-bit physical addressing with 16- bit registers.

???

Example of a College

- There are 500 students in a college. Each student has a unique number for identification 001 – 500.
- There are 8 courses in this college.
- Given a student id will it be possible to guess the course he has enrolled for? (without looking at the database)
  - Example: student id 246

Example of a College

- Assume each course can handle only 100 students.
- The roll numbers of students are assigned from 00 – 99 for each course
- A course id is maintained 1 – 8.
- The student id is now assigned as xyy where x is the course id and yy is the class roll number.
  - Example: 685

Example of a College

- Consider a database that allows you to enter two digits for of student id
- First enter → 06 (Course Id) then enter → 85 (class roll no)
- The first digit of the course id is always going to be 0, but it cannot be avoided because you have to enter two digits

Problem with 20 bit addresses

- 8086 addresses are 20-bits long. In a computer, 1 location is always 8 bits long.
- 8 bits = 1 byte
- 16 bits = 2 bytes
- 20 bits = 2.5 bytes
- Problem: 20 bits is not computer-compatible

Solution without Segmentation

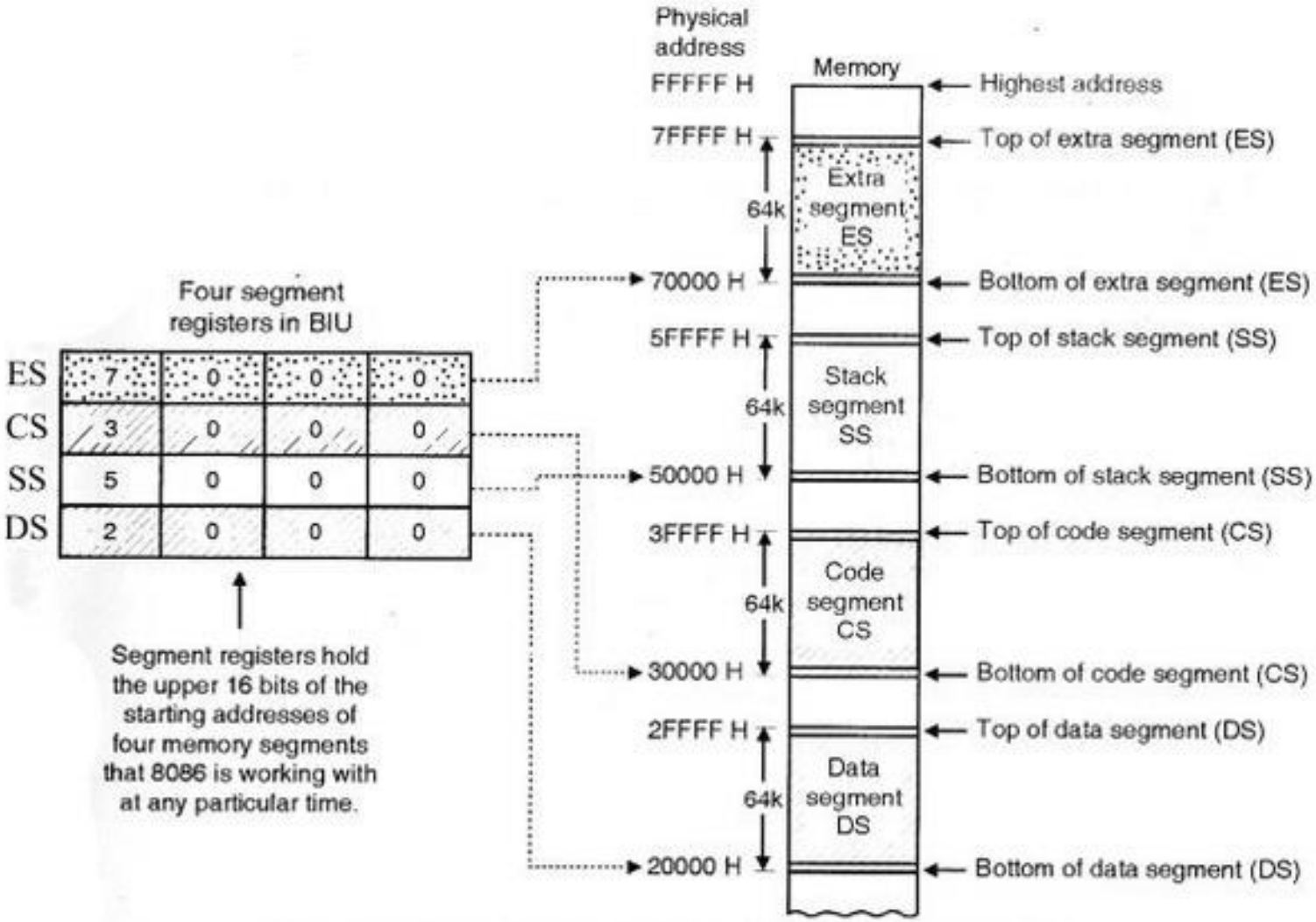
- Solution: use 24 bits = 3 bytes (with 4 bits always set to 0000)
- Again problem: huge amounts of data transfer wastage

The real need for Segmentation …

- Who created these problems? → WE
- Why did we create these problems? → We are not happy with 2^{16} = 64 KB memory
- We want to access 1 MB memory with 16 bits!

HOW ???

Memory Segmentation in 8086



One way of positioning four 64k byte segments within the 1M byte memory space of an 8086

What is Memory Segmentation ?

- Access a 20-bit address using 16-bits!
- Split the 20-bit address into two parts
  - Segment address (16-bits)
  - Offset address (16-bits)
- Regain the original physical address with the help of a small calculation
  - **P.A. = Segment address * 10 + offset address**
- Note : Segmentation is not optional for the 8086 architecture, it is a necessity.

Who will create these segments ?

PROGRAMMER

- Programmer creates these segments, but the Processor manages them.
- When a program is loaded, processor updates the segment registers with the start addresses of the segments in use.
- An offset register starts from 0000

Maximum Size of a Segment

- Offset register is 16-bits
  - Always starts at → **0000**
  - Last address possible → **FFFF**

- Bytes between FFFF – 0000 = 64KB (2^{16})

- Maximum size of a segment = **64 KB**

Minimum Size of a Segment

- Segment Registers store the Most Significant 16 bits of the 20-bit physical address
- The Least Significant 4 bits are not recorded.
- Hence 8086 imposes that a segment can start only at such a location whose address is a multiple of 10
- Example: 2000**0** or 5003**0** or 1234**0**
- It can never start at 12345 or 5003A, etc.

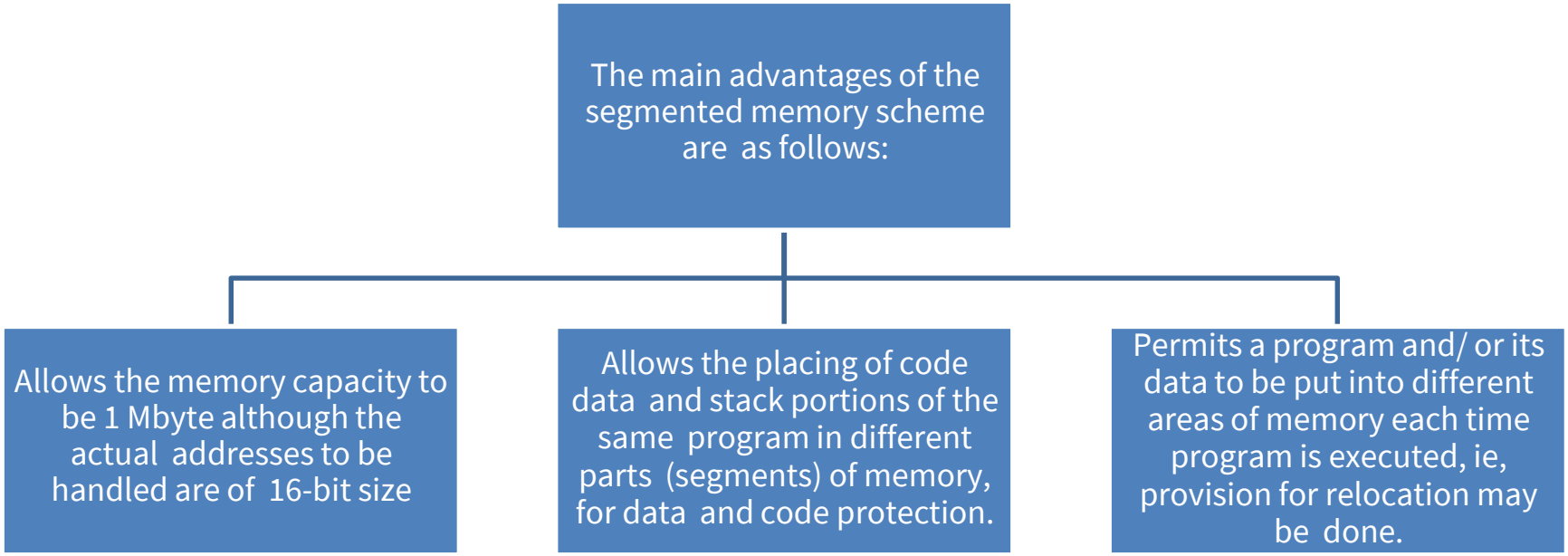
Minimum Size of a Segment

- Assume a segment starts at → 51230
- Since a segment cannot start at → 51231 , 51232 …5123A, … 5123F
- The next location available is → 51240
- i.e., 51240h – 51230h = 10h = 16 bytes
- Minimum size of a segment = **16 bytes**

Segment : Offset

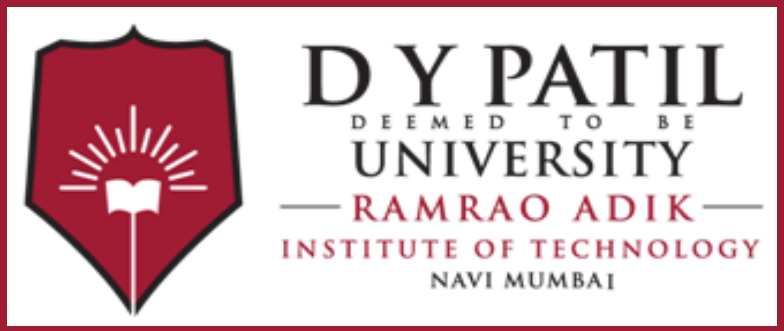
SEGMENT	SEGMENT REGISTER	OFFSET REGISTER
Code Segment	CSR	**Instruction Pointer (IP)**
Data Segment	DSR	**Source Index (SI)**
Extra Segment	ESR	**Destination Index (DI)**
Stack Segment	SSR	**Stack Pointer (SP) / Base Pointer (BP)**

Advantages of Segmentation



References

- Barry B. Brey, “Intel Microprocessors”, 8thEdition, Pearson Education India
- John Uffenbeck, “8086/8088 family: Design Programming and Interfacing”, PHI.
- Yu-Cheng Liu, Glenn A. Gibson, “Microcomputer System: The 8086/8088 Family, Architecture, Programming and Design”, Prentice Hall
- K. M. Bhurchandani and A. K. Ray, “Advanced Microprocessors and Peripherals”, McGraw Hill
- Douglas Hall, “Microprocessor and Interfacing”, Tata McGraw Hill.


D Y PATIL
DEEMED TO BE
UNIVERSITY
RAMRAO ADIK
INSTITUTE OF TECHNOLOGY
NAVI MUMBAI

Thank You