

Microprocessor (TE-SEM-V)

Compiled, reviewed and edited by:

Mrs. Vaishali Nirgude

TCET Mumbai

PREFACE

This book proffers a detailed study of the x86 family of microprocessors. The x86 family comprises Intel's 8086, 80186, 80286, 80386, 80486 and 80586 (Pentium) processors. The approach is to study the x86 family architecture based on the architecture of the elementary processor, i.e. the 8086. The higher-order processors are discussed based on the enhancements, improvements and differences the basic 8086. This is the best approach to learning the family architecture and it is followed by students worldwide. Most PCs across the world use the x86 architecture. Hence, it is an important subject that is taught and learnt at the academic and at the professional level.

Microprocessors form a key subject of study at the bachelor's level degree program of engineering, where it is taught as a core subject for all circuit-related branches, i.e. electronics, electrical, computer science and information technology. A prerequisite for mastering this subject is a course on logic design, implying that students need to know the basic building blocks of a digital system. A course on computer organization and architecture would be helpful to the student, but it is not mandatory for understanding the subject. However, not all institutions deal with computer architecture in their study modules before teaching microprocessors. Hence, this book is aimed at being the first introduction to microprocessors.

The theme of the book is centered around the architecture of the x86 microprocessor and a detailed study of assembly language programming and interfacing to external chips. Throughout the book, the emphasis is on ensuring that the reader can grasp concepts and ideas easily. To this end, solved examples, worked-out problems, tested programs and explanatory diagrams have been included. Microprocessor based system design involves interfacing of the processor with one or more peripheral devices for the purpose of communication with various input and output devices connected to it. During the early days of the microprocessor revolution, these techniques required complex hardware consisting of Medium scale integration devices making the design highly complex and time consuming. So, the manufacturers (INTEL) have developed a large number of general and special purpose peripheral devices most of them being single chip circuits. They are also programmable devices. Hence these peripheral devices are found to be of tremendous use to a system designer.

Peripheral devices, can broadly be classified into two categories.

(a) General purpose peripherals and

(b) Special purpose peripherals (Dedicated function peripherals)

General purpose peripherals are devices that perform a task but may be used for interfacing a variety of I/O devices to microprocessor. The general purpose devices are given below:

- Simple I/O -- (Non-programmable)
- ☐ Programmable peripheral Interface (PPI) – (8255)
- ☐ Programmable Interrupt Controller – (8259)
- ☐ Programmable DMA Controller – (8237/8257)
- ☐ Programmable Communication Interface – (8251)
- ☐ Programmable Interval Timer – (8253/8254)

Special function peripherals are devices that may be used for interfacing a microprocessor to a specific type of I/O device. These peripherals are more complex and therefore, relatively more expensive than general purpose peripherals.

General Guidelines for Students

1. Resource book is for structured and guided teaching learning process and therefore students are recommended to come with the same in every lecture.
2. Teaching will be done on the basis of resource book and home assignments will be covered from the same for the benefit of the students.
3. Resource book is framed to improve the academic result and therefore the students are recommended to take up all the module contents, home assignments and exercise seriously.
4. A separate notebook should be maintained for every subject.
5. Lectures should be attended regularly. In case of absence topic done in the class should be referred from the module before attending the next lecture.
6. Motivation, weightage and pre-requisite in every chapter have been included in order to maintain continuity and improve the understanding of the content to clarify topic requirement from exam point of view.
7. For any other additional point related to the topic instructions will be given by the subject teacher from time to time.

Microprocessor

SUBJECT RELATED

1. Weightage in the paper is 30-40% system designing and 70-60% theory and programming.
2. Questions are expected from all modules and students are instructed not to leave any module in option.
3. Weightage for Term Work – 25 marks, Practical and Oral -25 and Theory -75 marks. Importance should be given to term work for improving overall percentage in university examination.
4. Practice questions and university questions should be solved sincerely in order to enhance confidence level and to excel in university examination.
5. Definitions, key notations, solved examples and system design should be referred thoroughly from the modules after lecture session.

EXAM SPECIFIC

1. All modules are equally important. Emphasis on module 1, 2,3 and 4 can be given from the examination point of view because it covers (almost 60%) of university question paper which includes theory and system designing problem.
2. Neat labeled diagram should be drawn as per the requirement of the question mentioned in the question paper.
3. Read the question paper thoroughly first then choose the five questions. Attempt the one that you know the best first but do not change the internal sequence of the sub questions.
4. Minimum passing marks in theory paper - 30/75 and in term work 10/25.
5. For further subject clarification/ doubt in the subject, students can contact the subject teacher.

Guidelines for Writing Quality Answer

Theory:

1. Write content as per marks distribution.
2. Highlight the main points.
3. Write necessary content related to the point.
4. Draw neat and labeled diagrams wherever necessary.
5. While writing distinguishing points, write double the number of points as per the marks given, excluding the example.

Numerical:

Important steps should be written as they carry stepwise marks. The steps are as follows:

1. Given data
2. Diagrams (wherever applicable)
3. Formula (wherever applicable)
4. Pass by pass evaluation

Note- To get better result, we recommend that the quality of answers should be as per those given in university question-sample answers.

Syllabus Detailing

A. Keywords

	Sr.No.	Remembering	Understanding	Applying	Analyzing	Evaluating	Creating
FE	1	Label	Compare	Change	Conclude	Choose	Arrange
	2	List	Explain	Use	Deduce	Test	Collect
	3	Select	Illustrate	Show	Question	Revise	Modify
	4	Name	Classify	Complete	Illustrate	Evaluate	Rewrite
	5	State	Derive	Calculate	Outline	Determine	Create
	6	Write	Differentiate	Classify	Identify	Contrast	Construct
	7	Read	Discuss	Illustrate			Revise
SE	1	Outline	Convert	Compute	Differentiate	Explain	Solve
	2	Indicate	Give examples	Relate	Summarize	Relate	Design
	3	Describe	Express	Solve	Diagram	Select	Develop
	4	Define	Predict	Choose	Determine	Estimate	Draw
	5	Draw		Interpret	Categorize		Explain
TE	1	Recall	Distinguish	Review	Select	Compare	Specify
	2	Relate	Describe	Sketch	Experiment	Justify	Integrate
	3	Reproduce	Comprehend	Apply	Analyze	Describe	
	4	Find		Examine	Relate	Assess	
	5	Characterize		Schedule	Distinguish		
**BE	1	Cite	Summarize	All the above	Compare,	Conclude	Synthesize
	2	Match	Estimate		Contrast	Measure	
	3	Tabulate	Contrast		Investigate	Summarize	

B. Course Scheme

B.E. (Computer Engineering)	T.E. SEM: V
------------------------------------	--------------------

Course Name: Microprocessor V							Course Code:			
Teaching Scheme (Program Specific)					Examination Scheme (Formative/ Summative)					
Modes of Teaching / Learning / Weightage					Modes of Continuous Assessment / Evaluation					
Hours Per Week					Theory (100)		Practical/Oral (25)	Term Work (25)	Total	
Theory	Tutorial	Practical	Contact Hours	Credits	IA	ESE	PR/OR	TW	125	
3	-	2	5	4	25	75	-	25		
IA: In-Semester Assessment - Paper Duration – 1.5 Hours										
ESE: End Semester Examination - Paper Duration - 3 Hours										
The weightage of marks for continuous evaluation of Term work/Report: Formative (40%), Timely completion of practical (40%) and Attendance / Learning Attitude (20%)										
Prerequisite: Basic Mathematics										

C. Course Objectives and Course outcomes

Course Objective: The course intends to introduce basic and advanced software and hardware architecture of Intel X86 processors, use of assembly language and mixed mode programming. It also introduces microcontroller and its applications.

Course Outcomes: Upon completion of the course students will be able to:

Sr. No.	Course Outcomes	Cognitive levels of attainment as per Bloom's Taxonomy
1	Describe 16-bit architecture of 8086 Microprocessor.	L1, L2
2	Apply the assembly and mixed language programming to develop small embedded application.	L1, L2, L3
3	Sketch 8086 based system using memory and peripheral chips.	L1,L2, L3
4	Analyze the role of 32bit microprocessor architecture over 16 bit architecture.	L1,L2,L3,L4
5	Compare Pentium family microprocessors.	L1, L2, L3,L4
6	Differentiate between microprocessor and microcontroller.	L1, L2,L3,L4

D. Syllabus Detailing and Learning objectives

Module	Chapter	Detailed Content	Syllabus Detailing	Learning Objectives
Module 1	CH 1 Intel 8086 Microprocessor (Hours -8)	Architecture of 8086 processor , Register set, Memory segmentation, Functional Pin Diagram,	Purpose: To make students understand the roll of microprocessor in computer based system and study basic software and hardware architecture of 16 bit 8086/8088 microprocessors and their comparison .	1. To Describe the features, software architecture (Programmer's model / Register set) and Hardware architecture (Block Diagram) of Intel 8086 / 8088 processor. (R)

		Operating Modes, Minimum mode 8086 system and Timing diagrams , Maximum mode 8086 system and Timing diagrams.	<p>Distinguish between microprocessor and microcontroller. Understand the different operating modes of 8086 processor. Discuss the concepts of memory segmentation and interrupts.</p> <p>Scope –</p> <p>1. Academic Aspects- Study of basic architecture and features of 16 bit microprocessor 8086/8088.</p> <p>2. Technology Aspect- Understand the role of microprocessor in microprocessor based system and how it is useful from basic operations to large calculations.</p> <p>3. Application Aspect- Differentiate between microprocessor and microcontroller. How the microprocessor used for general purpose applications and microcontroller for specific applications.</p>	<p>2. To Distinguish between 8086 and 8088 processor. (U)</p> <p>3. Use of memory segmentation with its pros and cons. (A)</p> <p>4. Describe the interrupt structure and how 8086 respond to interrupt (U).</p> <p>5. Draw the minimum mode CPU configuration and memory cycles (R).</p> <p>6. Draw the maximum mode CPU configuration for 8086 and the functions of Intel 8288 bus controller. (R).</p>
			<p>Students Evaluation –</p> <p>1. Subjective questions on 8086 architecture, pin configuration, Programming Model, Memory Segmentation</p> <p>2. Lab experiments: Implement Five different experiments using assembly language for 8086 architecture.</p> <p>3. Viva questions on architecture, pin configuration, difference between 8086 and 8088 etc.</p>	

			4. Class test based on module1 and 2. 5. Lab test based on module	
Module 2	CH 2 Instruction set and Assembly Language Programming (Hours -06)	Instruction set, Addressing Modes, Assembler Directives, Macros and Procedure, Assembly Language Programming, Mixed Mode programming	<p>Purpose- This chapter focused on how do programming in Assembly language. Study of the different types of instruction set, assembler directives and addressing modes. Understand the different features and applications of ALP. How to use macro & procedure in programs. Differentiation between macros & procedure/function/subroutine.</p> <p>Scope –</p> <p>1. Academic Aspects- To understand, how to write program in assembly language for different applications.</p> <p>2. Technology Aspect- Understand the different features of ALP such as addressing modes, assembler directives, macros , procedures , arrays and string operations etc and Write macros as and when required to increase readability, productivity of program</p> <p>3. Application Aspect-</p> <p>Understand the basics and applications of ALP for Embedded system project development purpose.</p> <p>Students Evaluation –</p> <p>1. Subjective questions on addressing modes and programming model.</p>	<p>1. Explain all the instructions with examples, formats and their use in the programming. (U)</p> <p>2. Write assembly language program for 8086 using instruction set, assembler directives and addressing modes etc.(R)</p> <p>3. Assess assembly language programs using TASM/MASM which are run on 8086 microprocessor system (E)</p> <p>4.To Compare between Macros and procedure and Specify when to use macro and procedure .(E)</p> <p>5. To define macros as and when required to increase readability, productivity of program. (R)</p> <p>6. Experiment how to use C/C++ features in assembly language to make program for interactive and user friendly i.e . mixed mode programming (AN)</p>

			<p>2. Lab experiments: Implement Five different experiments using assembly language for 8086 architecture.</p> <p>3. Viva questions on macros, procedures, assembler directives and instruction set etc.</p> <p>4. Class test based on module1 and 2.</p> <p>5. Lab test based on module</p>	
Module 3	<p>CH 3 -</p> <p>Memory and Peripheral Interfacing with 8086</p> <p>(Hours -08)</p>	<p>Memory Interfacing - RAM and ROM</p> <p>8259 PIC – Interrupt, Types of Interrupts, Interrupt Service Routine, Interrupt Vector Table, Block Diagram of 8259, Interfacing the 8259 in single and cascaded mode with 8086.</p> <p>8255 PPI - Block diagram, Command word format, Interfacing 8255 with 8086</p>	<p>Purpose –To provide the students with the knowledge of various memory and peripheral devices which are interfaced with microprocessors. Understand the features, applications, control word formats of Intel 8255, 8253, 8254, 8237 and how to interface these peripheral devices with 8086/8088 in minimum and maximum mode and how to make them programmable.</p> <p>Scope –</p> <p>1. Academic Aspects-</p> <p>Design microprocessor based system for the given specifications.</p> <p>2. Technology Aspect-</p> <p>How to interface different peripheral devices and make them programmable for different applications. 3.</p> <p>Application Aspect-</p> <p>Use for real time applications such as square wave generation, traffic monitoring etc.</p>	<p>1. Describe the working of 8259PIC. How to initialize and make 8259 Programmable to accept software and hardware interrupt requests in different operating modes .Also Illustrate the working of cascade mode. (U)</p> <p>2. To Analyze the working of 8255 PPI. How it is used for 8 or 16 bit data transfer to /from peripheral devices and processor. (AN)</p> <p>3. Illustrate the working of 8253 and show demonstration for different applications like Square wave generator, Interrupt on terminal, software trigger etc (A).</p> <p>4. Explain the functionality of coprocessor and how it communicate with 8086 processor for execution of floating point operations. (E)</p> <p>5. Distinguish between minimum and maximum mode and Understand the concept of ODD and EVEN memory bank and how to interface SRAM and DRAM with 8086/8088 in minimum and maximum mode. (U)</p>

			Student Evaluation - 1. Multiple choice questions based on peripheral devices. 2. Subjective questions based on 8255, 8253, 8359, 8257, 8087, 8089 etc. 3. Group activity: Design of microprocessor based system for given specification. 4. Mini project: Based on real time applications.	6. Design microprocessor based system for given specification using minimum mode and maximum mode. (C)
Module 4	CH 4 Intel 80386DX Processor (Hours -08)	Architecture of 80386DX processor, Register Organization, Operating Modes: Real Mode, Protected Mode And Virtual 8086 Mode, Protected mode Address Translation mechanism: Segmentation and Paging..	Purpose –To introduce 32 bit architecture to the students. Understand the different advanced features of Intel 80386Dx processor over 8- bit architecture. Study of Software and hardware architecture of Intel 80386Dx. Scope – 1. Academic Aspects- To learn the different advanced features of 32 bit architecture such as operating modes, memory management , paging , virtual memory , interrupts and multitasking. 2. Technology Aspect- Understand the different operating modes and how to switch from real mode to protected mode. 3. Application Aspect- To understand techniques for faster execution of instructions and improve speed of operation and performance of microprocessors.	1. List the advanced features of Intel 80386Dx processor such as ALU size, segmentation and paging , operating modes and protection mechanism etc. and compare the performance of 80386Dx with its predecessors (R) 2. Differentiate between Intel 80386Dx and 80386Sx w.r.t. address bus, data bus, cache memory , physical memory etc. (AN) 3. Describe software architecture (Programmer's model / Register set) and Hardware architecture (Block Diagram) of Intel 80386Dx processor (U) 4. Recall Logical, Linear and physical addresses and advantages and drawbacks of memory segmentation and paging mechanism. (R) 5. Compare different operating modes of Intel 80386Dx processor w.r.t. size of Physical Memory, segmentation and paging mechanism, enter and exit from operating modes etc. (E)

			Student Evaluation - 1. Questions based on Finite Automata and lexical analyzer 2. Lab experiments based on hand written lexical analyzer 3. Mini project: NFA to DFA 4. GATE questions based on lexical analyzer.	6. Give example what is need of protection mechanism in multitasking environment during segment and page level translation mechanism. (U)
Module 5	CH 5 Pentium Family processors (Hours -07)	Superscalar architecture, Super pipelining, Data flow architecture, Comparative study of Pentium family processors.	Purpose – At the completion of this module, students should be able to list the advanced features of Pentium processor. To understand the superscalar architecture, super pipelining and MESI protocol. Students able to differentiate between different Pentium family processors w.r.t. different parameters. Scope – 1. Academic Aspects- Compare the advanced features of Pentium processor w.r.t. its predecessors. Differentiate between Von Neumann architecture, Hardwired architecture and Data flow architecture. 2. Technology Aspect- Understand and use the advanced features for large computations. 3. Application Aspect- Improve the speed of operations and performance of microprocessors.	1. List the advanced features of Pentium-1 such as DIB, superscalar architecture, Cache system , Branch prediction logic etc. and also relate P-1 with Intel 80386Dx, 80486Dx etc. (R) 2. Describe the superscalar architecture (More than one ALU) and super pipeline operations and analyze the performance of processor. (U) 3. Distinguish between different Pentium family processors w.r.t. different parameters. (U) 4. Describe the features of multicore processors like i3, i5 and i7. (R) 5. Predict how flushing / discarding of instructions from large size queue problem is solve using Branch prediction logic. (U) 6. Recall cache subsystem and understand the concept of MESI protocol in code and data cache organization. (R)

			Student Evaluation – 1. Theory and viva questions on Integer and floating point pipeline stages, cache subsystem and Branch Prediction logic. 2. Lab experiment : Case study on Pentium family processors. 3. Assignment 3: Questions based on module 5.	
Module 6	CH 6 The Microcontroller 8051 (Hours -08)	Introduction to 8051 Microcontroller, Architecture, Pin configuration, Memory organization, Input /Output Ports, Serial communication, Interrupts	Purpose – Differnciate between microprocessor and Microcontroller. Discuss the 8051. The architecture of 8051 , Pin Configuration, Memory Organization, I/O ports and Serial Communications with Interrupt.	1. List the different features of 8051 controller.(R) 2. Distinguish between Microprocessor and Microcontroller.(U) 3. Explain the softare and hardware architecture of 8051. (E) 4.Explain Memory organization details of 8051.(E) 5. Describe the Pin diagram and various functionality of 8051. (U) 6. Use of interrupt in 8051. (A)

List of Practical/ Experiments:

Practical Number	Type of Experiment	Practical/ Experiment Topic	Hrs.	Cognitive levels of attainment as per Bloom's Taxonomy
1	Basic Experiments	Apply Assembly Language Programing to enter and display 8 bit & 16 bits number	2	L1, L2, L3
2		Apply Assembly Language Programing to covert HEX to BCD and BCD to HEX.	2	L1, L2, L3
3	Design Experiments	Apply Assembly Language Programing to perform addition and subtraction of two 16 bits numbers using macros and procedure. (Menu Based).	2	L1,L2,L3
4		Apply Assembly Language Programing to perform string operations. (i)Accept, (ii) Display, (iii) Concatenation (iv) Compare	2	L1,L2,L3
5		Make use of 8086 Trainer kits in: 1. Hexkey pad Mode 2. Serial Mode	4	L1,L2,L3
6		Illustrate Interfacing on Intel 8086 with 8255-Programmable Peripheral Interface.	2	L1,L2,L3,L4
7	Advanced Experiments	Apply Mixed Language Programing to design a calculator.	2	L1,L2,L3
8		Develop program to interface mouse driver/keyboard/printer drivers.	4	L1,L2,L3,L4
9	Mini/Minor Projects/ Seminar/	1. Demonstrate PC-to-PC Communication via RS-232 Serial Port. 2. Develop an application on Mixed mode programming. 3. Develop an application using Arduino Controller. 4. Develop an application using Raspberry-PI.	6	L1,L2,L3,L4
10	Case Studies/ Group Presentation	1. Compare Multicore processors i3,i5, i7. 2. Compare Von Neumann , Hardwired and Data flow architecture 3. Recent development in hardware components.	4	L1,L2,L3,L4
	Total Hours		30	

Module	Contents	Page number
1	Intel 8086 Microprocessor <ul style="list-style-type: none"> • Architecture of 8086 processor • Register set • Memory segmentation • Functional Pin Diagram • Operating Modes • Minimum mode 8086 system and Timing diagrams • Maximum mode 8086 system and Timing diagrams. <p>Objective Questions</p> <p>Short Questions</p> <p>Long Questions</p>	16-49
2	Instruction set and Assembly Language Programming <ul style="list-style-type: none"> • Instruction set • Addressing Modes • Assembler Directives • Macros and Procedure • Assembly Language Programming • Mixed Mode programming <p>Objective Questions</p> <p>Short Questions</p> <p>Long Questions</p>	50-68
3	Memory and Peripheral Interfacing with 8086 Memory Interfacing <ul style="list-style-type: none"> • Memory Interfacing - RAM and ROM • 8259 PIC – Interrupt, Types of Interrupts, Interrupt Service Routine, Interrupt Vector Table, Block Diagram of 8259, Interfacing the 8259 in single and cascaded mode with 8086. • 8255 PPI - Block diagram, Command word format, Interfacing 8255 with 8086. <p>Objective Questions</p> <p>Short Questions</p> <p>Long Questions</p>	69-100

4	Intel 80386DX Processor: <ul style="list-style-type: none"> • Architecture of 80386DX processor • Register Organization • Operating Modes: Real Mode, Protected Mode And Virtual 8086 Mode • Protected mode Address Translation mechanism: Segmentation and Paging <p>Objective Questions with answers.....</p> <p>Short Questions</p> <p>Long Questions</p>	101-138
5	Pentium Family processors <ul style="list-style-type: none"> • Superscalar architecture, Super pipelining, • Data flow architecture • Comparative study of Pentium family processors. <p>Objective Questions with answers.....</p> <p>Short Questions</p> <p>Long Questions</p>	139-160
6	The Microcontroller 8051 <ul style="list-style-type: none"> • Introduction to 8051 Microcontroller • Architecture • Pin configuration • Memory organization • Input /Output Ports • Serial communication • Interrupts <p>Objective Questions with answers.....</p> <p>Short Questions</p> <p>Long Questions</p>	161-186

Module: 1

Intel 8086Architecture

1.1. Motivation:

At the completion of the Module, students should be able to understand:

- 16- bit microprocessor of Intel family.
- Software & Hardware architecture of Intel 8086 processor.
- Pipelined architecture

1.2. Syllabus:

Lecture	Content	Duration	Self Study
1	Architecture of 8086 processor	2 Lecture	3 hours
2	Register set	1 Lecture	2 hours
3	Memory segmentation	1 Lecture	2 hours
4	Functional Pin Diagram	1 Lecture	2 hours
5	Operating Modes	1 Lecture	2 hours
6	Minimum mode 8086 system and Timing diagrams	1 Lecture	2 hours
7	Maximum mode 8086 system and Timing diagrams	1 Lecture	2 hours

1.3. Weightage: 20 Marks

1.4. Learning Objectives: Students should be able to-

- 1 To Describe the features, software architecture (Programmer's model / Register set) and Hardware architecture (Block Diagram) of Intel 8086 / 8088 processor. **(R)**
2. To Distinguish between 8086 and 8088 processor. **(U)**
3. Use of memory segmentation with its pros and cons. **(A)**
4. Describe the interrupt structure and how 8086 respond to interrupt **(U)**.

5. Draw the minimum mode CPU configuration and memory cycles **(R)**.
6. Draw the maximum mode CPU configuration for 8086 and the functions of Intel 8288 bus controller. **(R)**.

1.5. Theoretical Background:

There are many evolutions in the microprocessor design. In 1978, Intel came out with the 8086. The Intel 8086 is a 16-bit microprocessor, implemented in N-channel, depletion load, silicon gate technology (HMOS), and packaged it in a 40 pin dual in line package. Following are the features of 8086 Microprocessor.

Features of 8086 microprocessor –

- Intel 8086 was launched in 1978.
- It was the first 16-bit microprocessor.
- This microprocessor had major improvement over the execution speed of 8085.
- It is available as 40-pin Dual-Inline-Package (DIP).
- It is available in three versions:
 - 8086 (5 MHz)
 - 8086-2 (8 MHz)
 - 8086-1 (10 MHz)
- It consists of 29,000 transistors
- 8086 has a 20 bit address bus can access up to 220 memory locations (1 MB).
- It can support up to 64K I/O ports.
- It provides 14, 16 -bit registers.
- It has multiplexed address and data bus AD0- AD15 and A16 – A19.
- It requires single phase clock with 33% duty cycle to provide internal timing.
- 8086 is designed to operate in two modes, Minimum and Maximum.
- It can prefetches up to 6 instruction bytes from memory and queues them in order to speed up instruction execution.
- It requires +5V power supply.
- A 40 pin dual in line package
- **Minimum and Maximum Modes:**
- The minimum mode is selected by applying logic 1 to the MN / MX input pin. This is a single microprocessor configuration.

- The maximum mode is selected by applying logic 0 to the MN / MX input pin. This is a multi-microprocessors configuration.

1.6. Abbreviations:

- **BIU** : Bus interface unit
- **EU** : Execution unit
- **CS** : Code segment
- **DS** : Data segment
- **ES** : Extra segment
- **SS** : stack Segment
- **FIFO** : first in first out
- **SP** : stack Pointer
- **BP** : Base pointer
- **SI** : source index
- **DI** : destination index
- **INTR** : Interrupt Request
- **INTA** : Interrupt Acknowledge
- **NMI** : Non Mask able

1.7. Formulae: Nil

1.8. Key Definitions:

- 1) **Segmentation:** The 8086/88 has only 16-bit registers, its Mega (220) byte of address space is split into segments - logical units of memory that may be up to 64K (216) bytes long. Each segment is made up of contiguous memory locations and is an independent, separately addressable unit. Every segment is assigned (by software) a base address, which is its starting location in the memory space. Apart from having to begin on 16-byte memory boundaries, there are no restrictions on segment locations.
- 2) **Pipelining:** The process of fetching the next instruction, when the present instruction is being executed is called as pipelining.
- 3) **Registers:** Registers made of Flip Flop. The 80x86 family, like other processors, has both general and special purpose registers available to the assembly language programmer. These registers can be loosely classified into Data Registers, Address Registers, and Status Registers.
- 4) **Flags Register:** Determines the current state of the processor.
- 5) **Bus Interface Unit:**

1. It provides a full 16 bit bidirectional data bus and 20 bit address bus.
2. The bus interface unit is responsible for performing all external bus operations.

6) **Execution Unit:** The Execution unit is responsible for decoding and executing all instructions.

- Queue: Gives information about the status of the code-pre fetch queue
- Segmented memory: 8086 mp can access 1MB memory. This memory is logically divided into four segments. They are code segment, data segment, extra segment, stack segment. Size of each segment is 64Kb.
- Pipelined operation: 8086 BIU & EU works simultaneously. Fetching and executing an instruction is called pipelining.

1.9. Course Content:

Lecture: 1

8086 Microprocessor Architecture

Learning Objective: In this lecture student will able to learn about Hardware Architecture 8086/8088 Microprocessor.

1.9.1 The architecture of the IC 8086 :

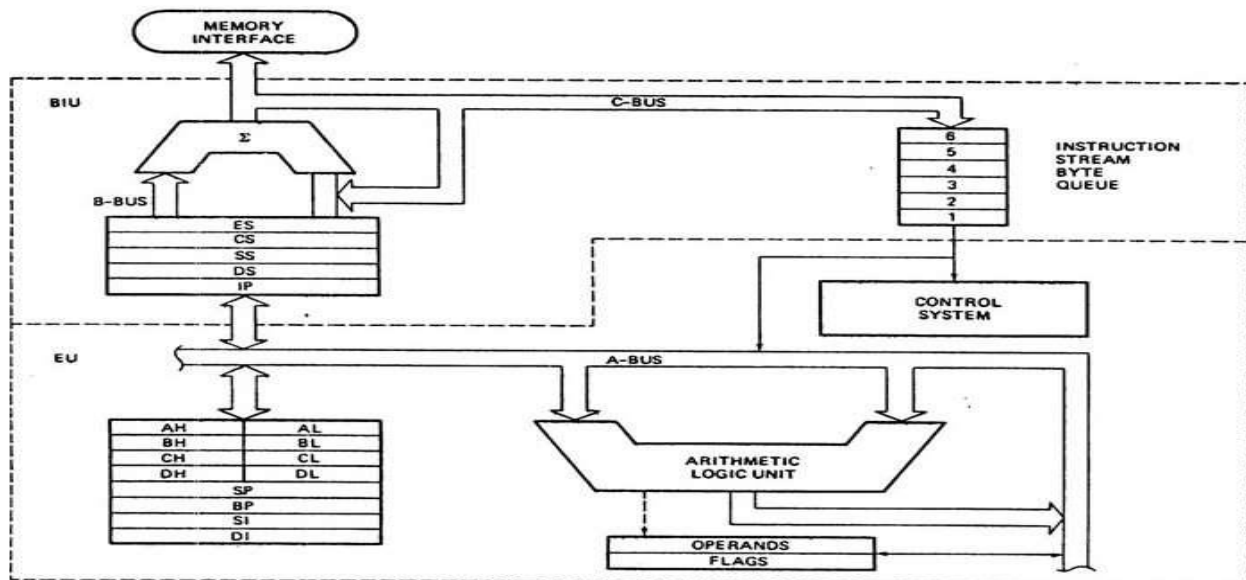


Fig 1.1 The architecture of the Intel 8086

1.9.2 The architecture of the IC 8088:

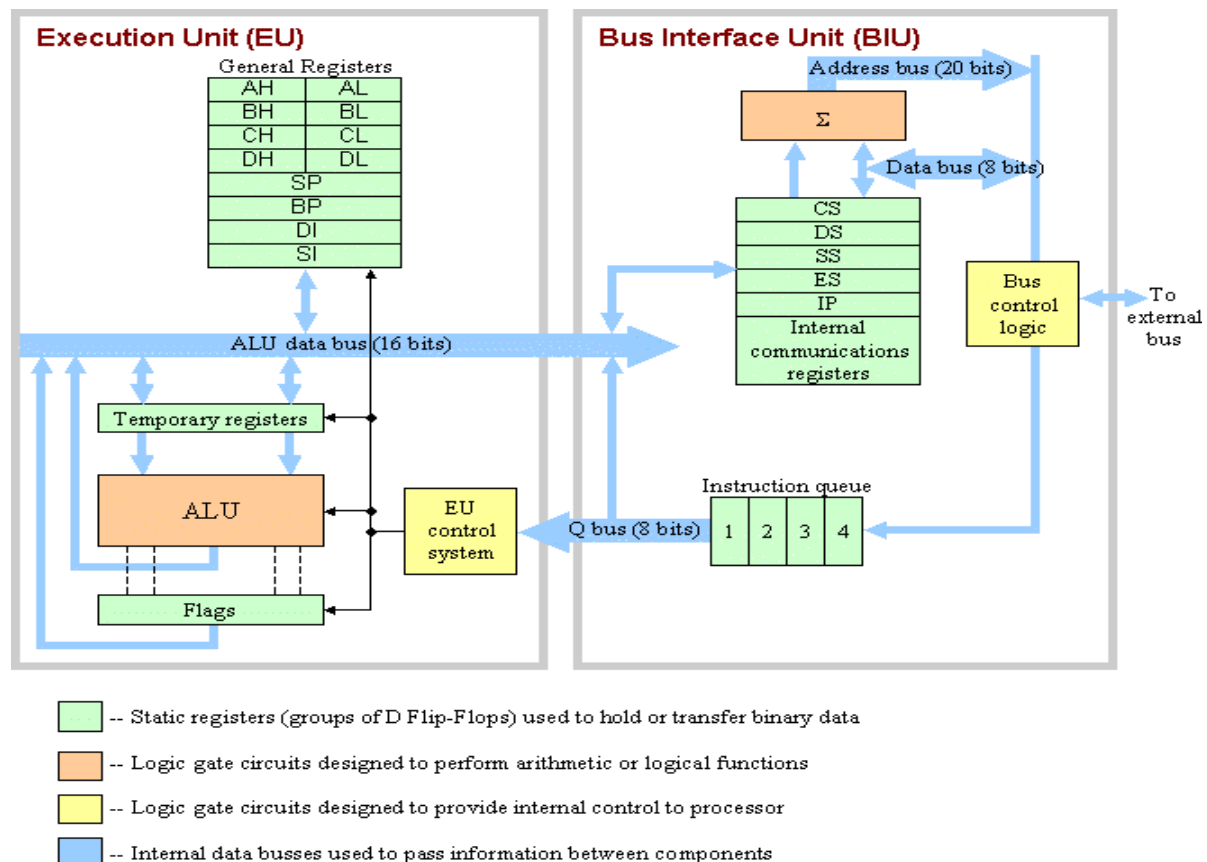


Fig. 1.2 The architecture of the IC 8088

Difference between 8086 and 8088 Microprocessors:

The only difference between an 8088 microprocessor and an 8086 microprocessor is the BIU. In the 8088, the BIU data bus path is 8 bits wide versus the 8086's 16-bit data bus. Another difference is that the 8088 instruction queue is four bytes long instead of six.

Table 1. Difference between 8086 and 8088 Microprocessors

8086	8088
It has 16 bit data bus. It has AD0-AD15 lines	It has 8 bit data bus. It has AD0-AD7 lines
It has BHE' signal to access higher byte	As data bus is 8-bit wide, it does not have BHE' signal

8086 requires memory banking to transfer 16 bit data at a time.	8088 does not require memory banking as it has an 8 bit data bus.
8086 performs faster memory operations as it can transfer 16 bits in one cycle.	8088 performs slower memory operations as it can transfer only 8 bits in one cycle.
8086 has a 6 byte pre-fetch queue for pipelining.	8088 has a 4 byte pre-fetch queue for pipelining.
8086 has an M/IO' pin to differentiate between memory and I/O operations.	8088 has an IO/M' pin to differentiate between memory and I/O operations.
8086 BIU will fetch new bytes into the pipelining queue when 2 bytes of the queue are empty.	8088 BIU will fetch a new byte into the pipelining queue when 1 byte of the queue is empty.

The 8086 CPU is divided into two independent functional units:

1. Bus Interface Unit (BIU)

2. Execution Unit (EU)

Bus Interface Unit (BIU)

The function of BIU is to:

- Fetch the instruction or data from memory.
- Write the data to memory.
- Write the data to the port.
- Read data from the port.

Instruction Queue

1. To increase the execution speed, BIU fetches as many as six instruction bytes ahead to time from memory.
2. All six bytes are then held in first in first out 6 byte register called instruction queue.
3. Then all bytes have to be given to EU one by one.
4. This pre fetching operation of BIU may be in parallel with execution operation of EU, which improves the speed execution of the instruction.

Execution Unit (EU)

The functions of execution unit are:

- To tell BIU where to fetch the instructions or data from.
- To decode the instructions.
- To execute the instructions.
- The EU contains the control circuitry to perform various internal operations. A decoder in EU decodes the instruction fetched memory to generate different internal or external control signals required to perform the operation.

- EU has 16-bit ALU, which can perform arithmetic and logical operations on 8-bit as well as 16-bit.

Let's check the take away from this lecture

1) Which of the processor listed below have 16-bit internal data registers?

- i) 8086 ii) 80286
iii) 80386 iv) 8088

2) 8088 microprocessor differs with 8086 microprocessor in

- i) Data width on the output ii) Address capability
iii) Support of coprocessor iv) Support of MAX / MIN mode

L4. Exercise:

Q.1 Explain the architecture of the IC 8086 with a neat block diagram

Q.2 Differentiate between: BIU and EU of 8086 microprocessor.

Questions/problems for practice:

Q. 3 List the features of 8086/8088 processor

Learning from the lecture '8086/8088 Microprocessor Architecture':

Student will be able to describe Hardware Architecture 8086/8088 Microprocessor.

Lecture: 2

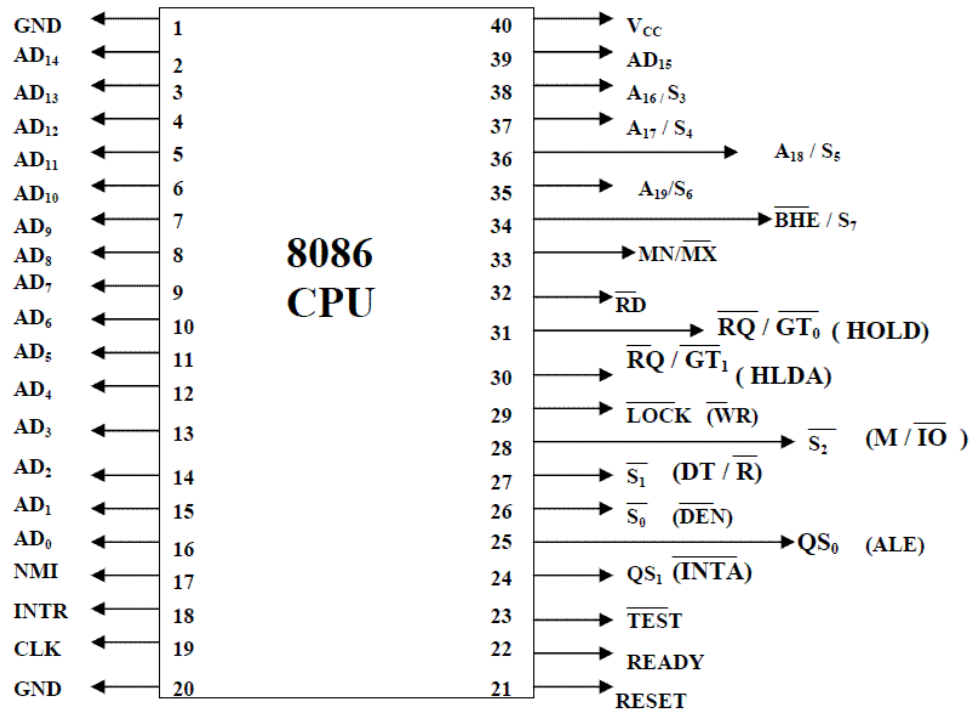
Pin Configuration

Learning objective: In this lecture students will be able to understand the pin configuration of 8086 microprocessor in minimum mode and maximum mode

1.9.3 Pin Diagram of 8086: The microprocessor 8086 is a 16-bit CPU available in three clock rates, i.e. 5, 8 and 10 MHz, packaged in a 40 pin Cerdip or plastic package. The 8086 operates in single processor or multiprocessor configurations to achieve high performance. The pin configuration is shown in Fig. 1.1. Some of the pins serve a particular function in minimum mode (single processor mode) and others function in maximum mode (multiprocessor mode) configuration.

The 8086 signals can be categorized in three groups. The first are the signals having common functions in minimum as well as maximum mode, the second are the signals which have special functions for minimum mode and the third are the signals having special functions for maximum mode.

Pin Diagram of 8086



MN/MX- is an input pin used to select one of this mode .when **MN/MX is high** the 8086 operates in minimum mode .In this mode the 8086 is configured to support small single processor system using a few devices that the system bus .when **MN/MX is low** 8086 is configured to support multiprocessor system.

The **AD0-AD15** lines are a 16bit multiplexed addressed or data bus. During the 1st clock cycle AD0-AD15 are the low order 16Bit adders. The 8086 has a total of 20 address line ,the upper 4 lines are multiplexed with the state signal that is **A16/S3 , A17/S4 , A18/S5 , A19 /S6**.During the first clock period of a best cycle the entire 20bit address is available on these line. During all other clock cycles for memory and i/o operations AD15-AD0 contain the 16 bit data and **S3,S4,S5,S6** become the status line .S3 and S4 are decoded as follows

A17/S4 A16/S3 Function

S4	S3	Segment Register
0	0	Extra Segment
0	1	Stack Segment

1	0	Code or No segment
1	1	Data Segment

There for the 1st clock cycle of an instruction execution the A17/S4 And A16/S3 pins Specify which Segment register generate the segment portions of the 8086 address

BHE/S7 is used as **best high enable** during the 1st click cycle of an instruction execution .the BHE can be used in conjunction with AD0 to select the memory

RD is low when the data is read from memory or I/O location .

TEST is an input pin and is only used by the wait instruction .the 8086 enter a wait state after execution of the wait instruction until a low is Sean on the test pin.

INTR is a maskable interrupt input.

NIM is the non maskable interrupt input.

RESET is the system set reset input signal it terminates all the activities it clear **PSW,IP,DS,SS,ES** and the instruction Queue.

DT/R(Data Transmit or receive):is an o/p signal required in system that uses the data bus transceiver

ALE is an **address latch enable** . Is an o/p signal provided by the 8086 and can be used to demultiplexed AD0 to AD15 in to A10 toA15 and D0 to D15.

M/IO is an 8086 output signal to distinguish a memory access and i/o access.

WR is used by the 8086 for performing write memory or write i/o operation .

INTA(interrupt acknowledgement signal)

INTA is the interrupt acknowledgment signal

HOLD and HOLDA a high on the HOLD pin indicates that another master is required to take over the S/M bus

CLK clock provides the basic timing signals for the 8086 and bus controls

Let's check the take away from this lecture

3) In 8086 microprocessor one of the following statements is not true.

- a) Coprocessor is interfaced in MAX mode
- b) Coprocessor is interfaced in MIN mode
- c) I/O can be interfaced in MAX / MIN mode
- d) Supports pipelining

L5. Exercise

Q.4 Draw and explain Pin diagram of 8086.

Questions/Problems for practice:

Q.5 Explain the functions of following pins. HOLD and HLDA

Learning from this lecture 'Pin Configuration':

Student will able to state the functions of different pins used in 8086/8088 microprocessor.

Lecture: 3

Programming Model, Memory Segmentation

Learning Objective: In this lecture student will able to learn about Programming Model and Memory Segmentation 8086/8088 Microprocessor.

1.9.4 Software Architecture / Register Set/ Programmer's model of Intel 8086 Microprocessor:

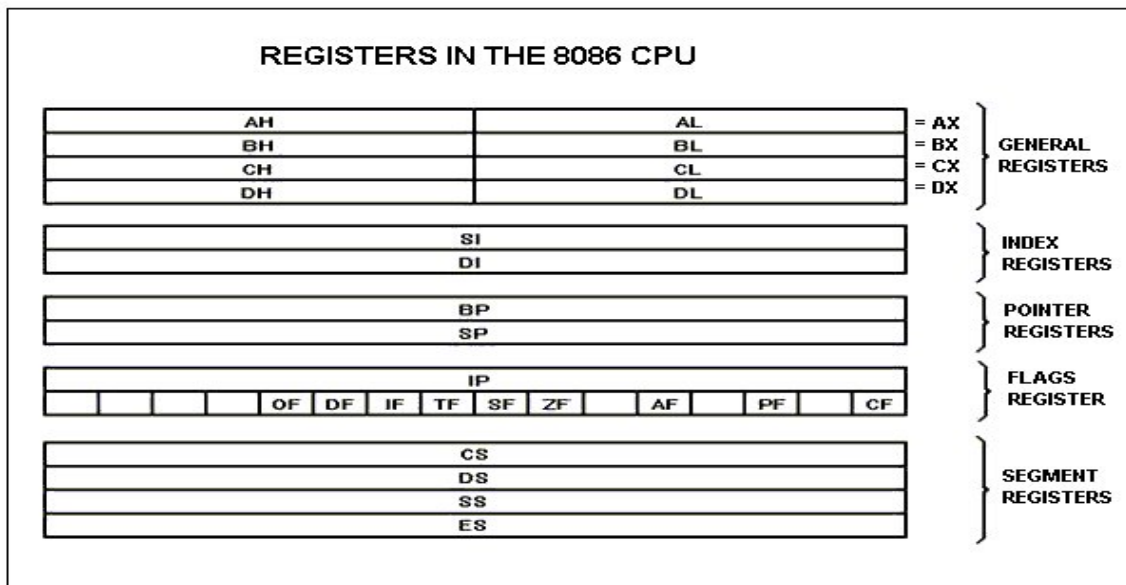


Fig. 1.3 Register Set

GENERAL PURPOSE REGISTERS

8086 CPU has 8 general purpose registers, each register has its own name:

AX - the accumulator register (divided into AH / AL):

1. Generates shortest machine code
2. Arithmetic, logic and data transfer
3. One number must be in AL or AX
4. Multiplication & Division
5. Input & Output

BX - the base address register (divided into BH / BL).

CX - the count register (divided into CH / CL):

1. Iterative code segments using the LOOP instruction
2. Repetitive operations on strings with the REP command
3. Count (in CL) of bits to shift and rotate

DX - the data register (divided into DH / DL):

1. DX:AX concatenated into 32-bit register for some MUL and DIV operations
2. Specifying ports in some IN and OUT operations

SI - source index register: [provides the address of the source data](#)

1. Can be used for pointer addressing of data
2. Used as **source** in some string processing instructions
3. Offset address relative to **DS**

DI - destination index register:

1. Can be used for pointer addressing of data
2. Used as **destination** in some string processing instructions
3. Offset address relative to **ES**

BP - base pointer:

1. Primarily used to access parameters passed via the stack
2. Offset address relative to SS

SP - stack pointer:

1. Always points to top item on the stack

2. Offset address relative to SS
3. Always points to word (byte at even address)
4. An empty stack will had $SP = FFFh$

SEGMENT REGISTERS

CS - points at the segment containing the current program.

DS - generally points at segment where variables are defined.

ES - extra segment register, it's up to a coder to define its usage.

SS - points at the segment containing the stack.

Flag Register of 8086:

- A flag is a flip-flop which indicates some condition produced by the execution of an instruction or controls certain operations of the EU.
- The Flag Register is a special register associated with the ALU.
- A 16-bit flag register in the EU contains nine active flags.
- Fig. shows the location of the nine flags in the flag register

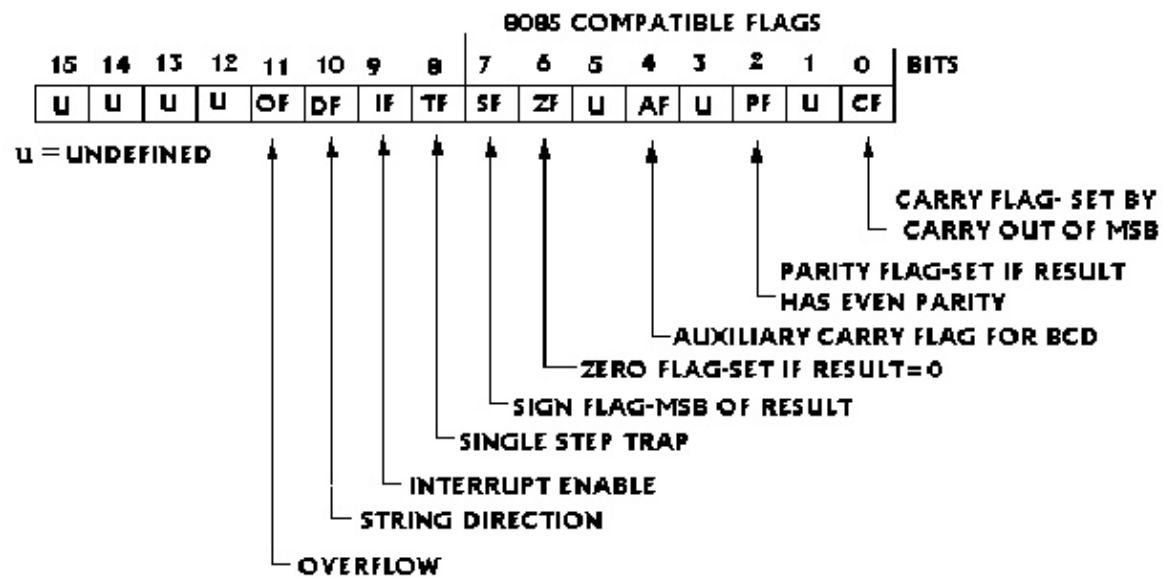


Fig.5 8086 flag register format

Fig. 1.4 Flag Register structure

Flags is a 16-bit register containing 9 1-bit flags:

- Overflow Flag (OF) - set if the result is too large positive number, or is too small negative number to fit into destination operand.
- Direction Flag (DF) - if set then string manipulation instructions will auto-decrement

index registers. If cleared then the index registers will be auto-incremented.

- Interrupt-enable Flag (IF) - setting this bit enables maskable interrupts.
- Single-step Flag (TF) - if set then single-step interrupt will occur after the next instruction.
- Sign Flag (SF) - set if the most significant bit of the result is set.
- Zero Flag (ZF) - set if the result is zero.
- Auxiliary carry Flag (AF) - set if there was a carry from or borrow to bits 0-3 in the AL register.
- Parity Flag (PF) - set if parity (the number of "1" bits) in the low-order byte of the result is even.
- Carry Flag (CF) - set if there was a carry from or borrow to the most significant bit during last result calculation.

1.9.5 Memory Segmentation:

Segment Registers:

- The physical address of the 8086 is 20-bits wide to access 1 Mbyte memory locations. However, its registers and memory locations which contain logical addresses are just 16-bits wide. Hence 8086 uses memory segmentation.
- It treats the 1 Mbyte of memory as divided into segments, with a maximum size of a segment as 64 Kbytes. Thus any location within the segment can be accessed using 16 bits.
- The 8086 allows only four active segments at a time, as shown in the Fig.
- For the selection of the four active segments the 16-bit segment registers are provided within the BIU of the 8086. These four registers are: Code segment (CS) register, the data segment (DS) register, the stack segment (SS) register, and the extra segment (ES) register. These are used to hold the upper 16-bits of the starting addresses of the four memory segments, on which 8086 works at a particular time.

Rules For Memory Segmentation:

1. The four segments can overlap for small programs. In a minimum system all four segments can start at title address 00000H.
2. The segment can begin/start at any memory address which is divisible by 16.

Advantages of Memory Segmentation:

1. It allows the memory addressing capacity to be 1 Mbyte even though the address associated with individual instruction is only 16-bit.
2. It allows instruction code, data, stack, and portion of program to be more than 64 KB long by using more than one code, data, stack segment, and extra segment.
3. It facilitates use of separate memory areas for program, data and stack.

4. It permits a program or its data to be put in different areas of memory, each time the program is executed i.e. program can be relocated which is very useful in multiprogramming.

Let's check the take away from this lecture

5. Which of the processor listed below have 16-bit internal data registers?

1. 8086
2. 8088
3. 80286
4. 80386

L6 Exercise:

Q.7 Explain register set of 8086 processor

Q.8 Draw structure of flag register

Q.9 What is segmented memory? Enumerate the advantages of segmented memory with reference to the 8086 microprocessor.

Questions/problems for practice:

Q.10 Explain memory segmentation concept of 8086?

Learning from the lecture Programming Model and Memory Segmentation':

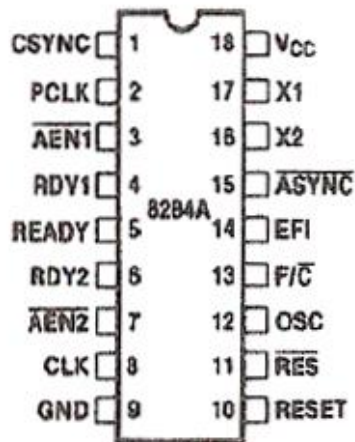
Student will able to define the concept Programming Model and Memory Segmentation.

Lecture: 4

Study of 8284 Clock Generator, Operating Modes

Learning Objective: In this lecture students will understand the different operating modes of 8086 processor.

1.9.6 Clock Generator (8284A): Signals



RESET Signals

- #RES Reset input: Active low. Usually connected to an RC circuit to provide automatic reset at power on.
- RESET output: Synchronized to Clk. Connect to the 8086/8088 RESET input.

READY Signals

- #AEN1 and #AEN2 address enable inputs: Used with RDY1 and RDY2 inputs to generate the READY output. The READY output is connected to the READY input on the 8086/8088 mP to control memory wait states.
- #ASYNC input: for READY output synchronization. Selects 1 or 2 stages of synchronization for the RDY1 and RDY2 inputs.

Clock Generator (8284A): Block Diagram

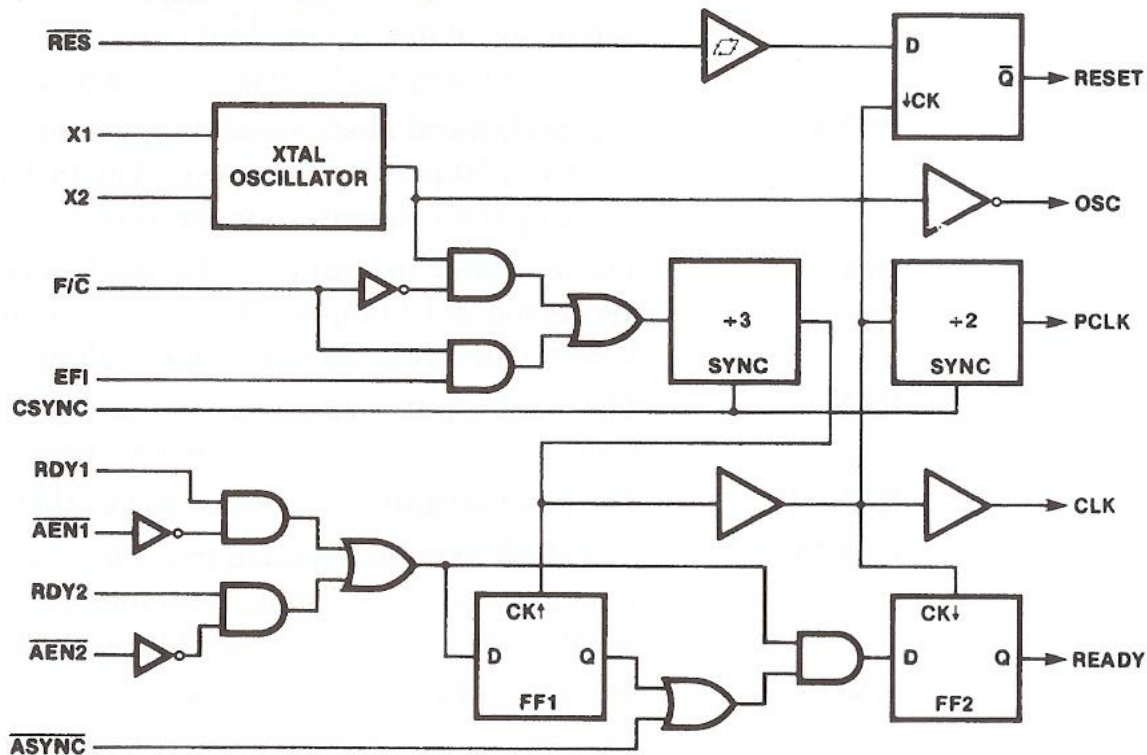


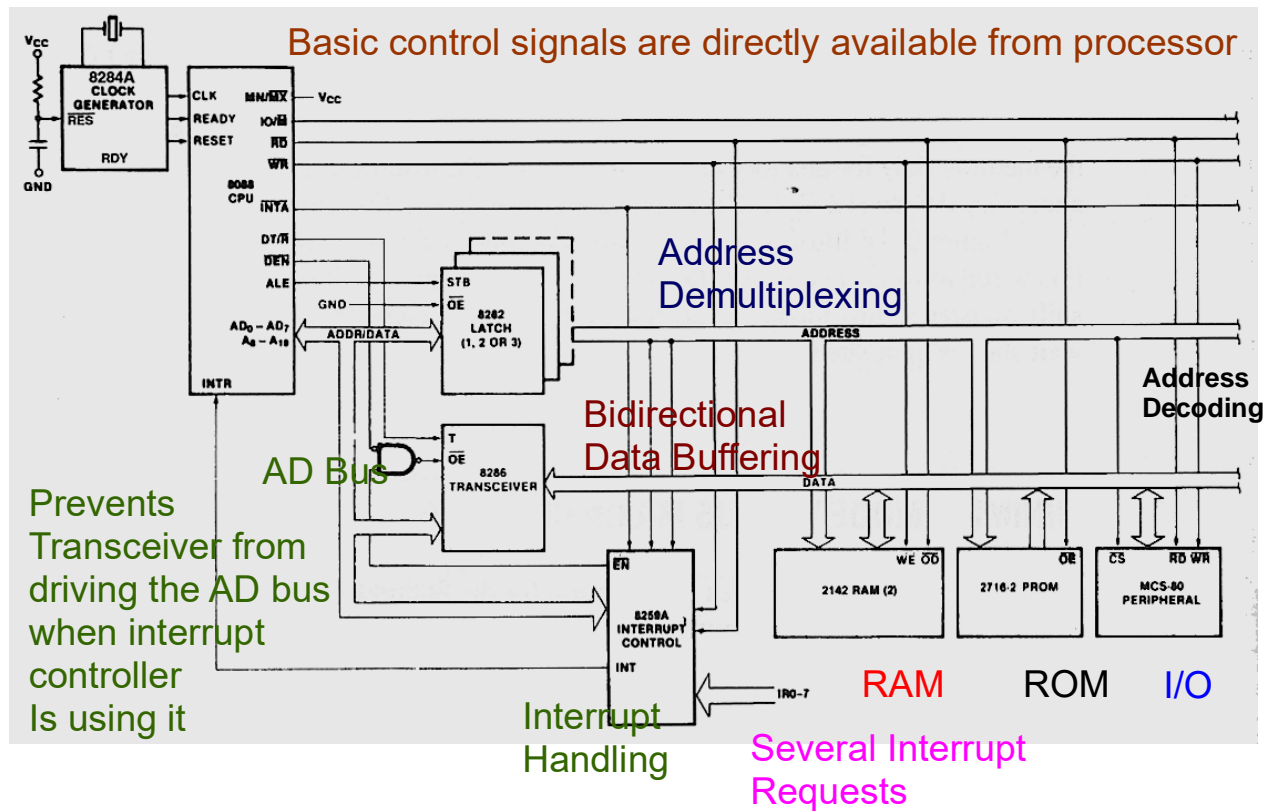
Fig. 1.5 Block diagram of 8284 Clock Generator

Operating Modes:

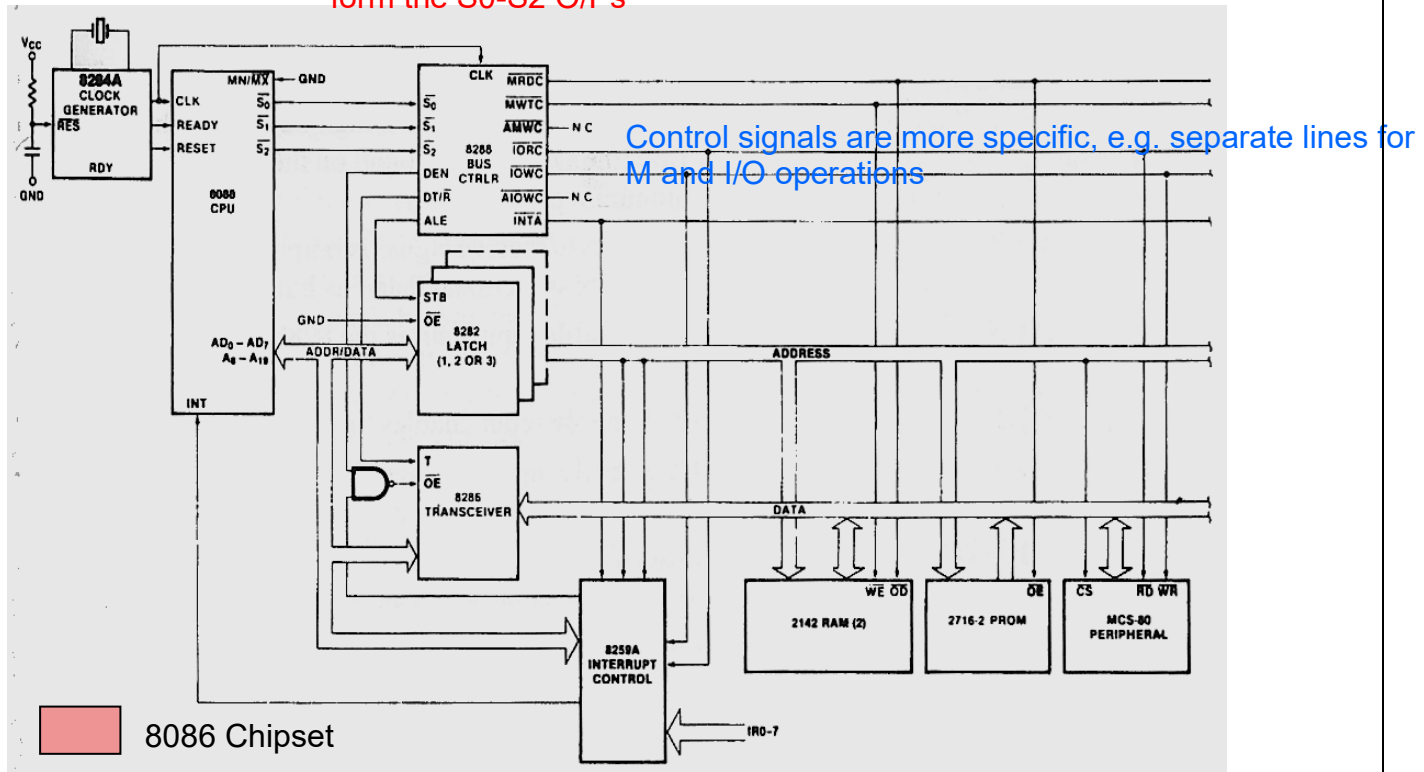
Minimum and Maximum Modes:

- MN/#MX input on 8088/8086 selects min (0V) or max (+V) mode
- Minimum mode is the least expensive way to configure a 8086/8088 system:
 - Bus control signals are generated directly by processor
 - Good backward compatibility with earlier 8085A 8 bit processor
 - Same control signals
 - Support same peripherals
- Maximum mode provides greater versatility at higher cost.
 - New control signals introduced to support 8087 coprocessor (e.g. QS0 & QS1) and multiprocessor operation (e.g. #RQ/GT0 & RQ/GT1)
 - Control signals omitted must be externally generated using an external bus controller, e.g. 8288. The controller decodes those control signals from the now compressed form of 3 control bits (#S0, #S1, #S2)
 - Can be used with the 8087 math coprocessor
 - Can be used with multiprocessor systems

A) Use of 8086 in the Minimum Mode:



B) Use of 8086 in the Maximum Mode:



Let's check the take away from this lecture

9. In 8086 microprocessor one of the following statements is not true.

- a) Coprocessor is interfaced in MAX mode
- b) Coprocessor is interfaced in MIN mode
- c) I/O can be interfaced in MAX / MIN mode
- d) Supports pipelining

L7 Exercise:

Q.12 Write short note on 8284 clock generator

Question/ problems for practice

Q.13 Draw and explain the CPU configuration of 8086 maximum mode

Q. 14 Draw and explain the CPU configuration of 8086 8086 minimum mode

Learning from the Lecture ‘ Study of 8284 Clock Generator, Operating Modes’:

Student will able to define the concept of Operating Modes

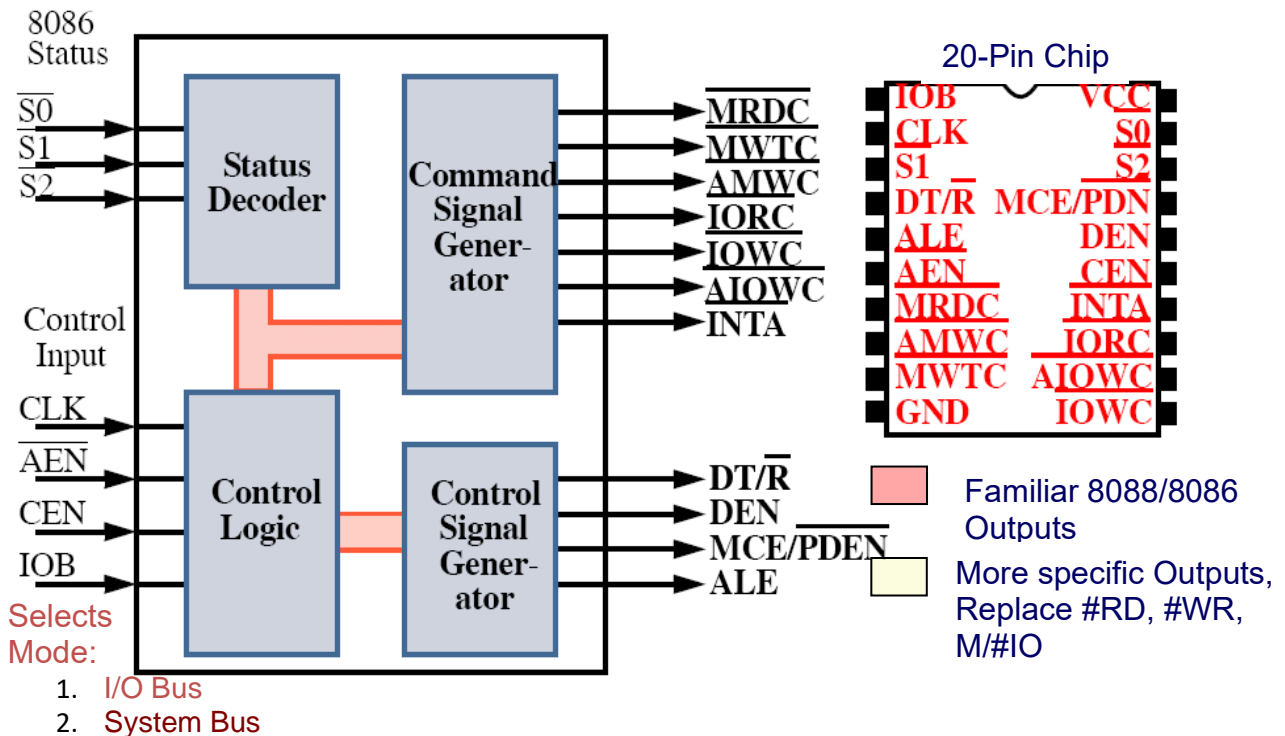
Lecture: 5

Study of 8288 Bus Controller

Learning Objective: In this lecture students will be able to understand 8288 Bus Controller

1.9.7 Intel Bus Controller 8288 :

8288 Bus Controller



- S0, S1, S2 inputs: Status bus bits from processor. Decoded by the 8288 to produce the normal control signals
- CLK input: From the 8284A clock generator
- ALE output: Address latch enable output for demuxing address/data
- DEN output: Data bus enable output to enable data bus buffer. Note opposite polarity to #DEN output in minimum mode.
- DT/#R output: Data transmit/Receive output to control direction of the bi directional data bus.
- #INTA output: Acknowledge a hardware interrupt applied to the INTR input of the processor.
- IOB input: I/O bus mode input. Selects operation in either I/O bus mode or system bus mode.
- #AEN input: Address Enable input. Used by the 8288 to enable memory control signals. Supplied by a bus arbiter in a multiprocessor system

- CEN input: Control Enable input. Enables the generation of command outputs from the 8288.
- #IORC output: Input/Output read control signal.
- #IOWC output: Input/Output write control signal.
- #AIOWC output: Advanced Input/Output control signal.
- #MRDC output: Memory read control signal.
- #MWTC output: Memory write control signal.
- #AMWT output: Advanced Memory write control signal.
- MCE/#PDEN output: Master cascade/Peripheral data output. Selects cascade operation if IOB=0 or enables I/O bus transceivers if IOB=5V

Let's check the take away from this lecture

L8 Exercise :

Q.15 Explain 8288 in detail i.e. clock logic, reset logic, ready logic.

Questions/problems for Practice

Q.16 Explain 8288 Bus controller

Learning from the lecture 'Study of 8288 Bus Controller':

Student will be able to understand the concept of 8288 Bus Controller

Lecture : 6 & 7

Timing diagrams for Read and Write operations

Learning Objective: In this lecture students will be able to learn how to draw Read and Write timing diagrams.

Let's check the take away from this lecture

1.9.8 Bus Timing diagrams

The 8086 has a combined address and data bus commonly referred to as a time multiplexed address and data bus. The main reason behind multiplexing address and data over the same pins is the maximum utilisation of processor pins and it facilitates the use of 40 pin standard DIP package. The bus can be demultiplexed using a few latches and transceivers, whenever required.

Basically, all the processor bus cycles consist of at least four clock cycles. These are referred to as T_1 , T_2 , T_3 and T_4 . The address is transmitted by the processor during T_1 . It is present on the bus only for one cycle. During T_2 , i.e. the next cycle, the bus is tristated for changing the direction of bus for the following data read cycle. The data transfer takes place during T_3 and T_4 . In case, an addressed device is slow and shows 'NOT READY' status the wait states T_w are inserted between T_3 and T_4 . These clock states during wait period are called idle states (T_i), wait states (T_w) or inactive states. The processor uses these cycles for internal housekeeping. The address latch enable (ALE) signal is emitted during T_1 by the processor (minimum mode) or the bus controller (maximum mode) depending upon the status of the $\overline{MN}/\overline{MX}$ input. The negative edge of this ALE pulse is used to separate the address and the data or status information. In maximum mode, the status lines \overline{S}_0 , \overline{S}_1 and \overline{S}_2 are used to indicate the type of operation. Status bits \overline{S}_3 to \overline{S}_7 are multiplexed with higher order address bits and the \overline{BHE} signal. Address is valid during T_1 while the status bits \overline{S}_3 to \overline{S}_7 are valid during T_2 through T_4 . The Fig.1.7 shows a general bus operation cycle of 8086.

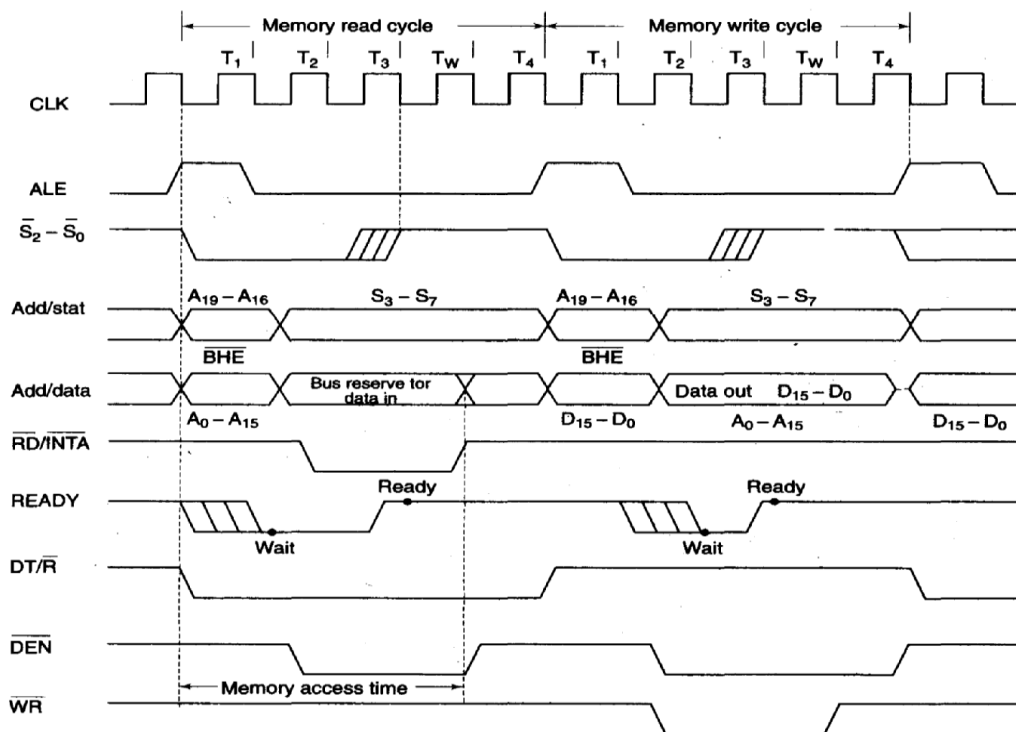


Fig. 1.7 General Bus Operation Cycle in Maximum Mode

MINIMUM MODE 8086 SYSTEM AND TIMINGS

In a minimum mode 8086 system, the microprocessor 8086 is operated in minimum mode by strapping its $\overline{MN}/\overline{MX}$ pin to logic 1. In this mode, all the control signals are given out by the microprocessor chip itself. There is a single microprocessor in the minimum mode system. The remaining components in the system are latches, transreceivers, clock generator, memory and I/O devices. Some type of chip selection logic may be required for selecting memory or I/O devices, depending upon the address map of the system.

The latches are generally buffered output D-type flip-flops, like, 74LS373 or 8282. They are used for separating the valid address from the multiplexed address/data signals and are controlled by the ALE signal generated by 8086. Transceivers are the bidirectional buffers and some times they are called as data amplifiers. They are required to separate the valid data from the time multiplexed address/data signal. They are controlled by two signals, namely, \overline{DEN} and DT/\overline{R} . The \overline{DEN} signal indicates that the valid data is available on the data bus, while DT/\overline{R} indicates the direction of data, i.e. from or to the processor. The system contains memory for the monitor and users program storage. Usually, EPROMS are used for monitor storage, while RAMs for users program storage. A system may contain I/O devices for communication with the processor as well as some special purpose I/O devices. The clock generator generates the clock from the crystal oscillator and then shapes it and divides to make it more precise so that it can be used as an accurate timing reference for the system. The clock generator also synchronizes some external signals with the system clock. The general system organisation is shown in Fig. 1.8. Since it has 20 address lines and 16 data lines, the 8086 CPU requires three octal address latches and two octal data buffers for the complete address and data separation.

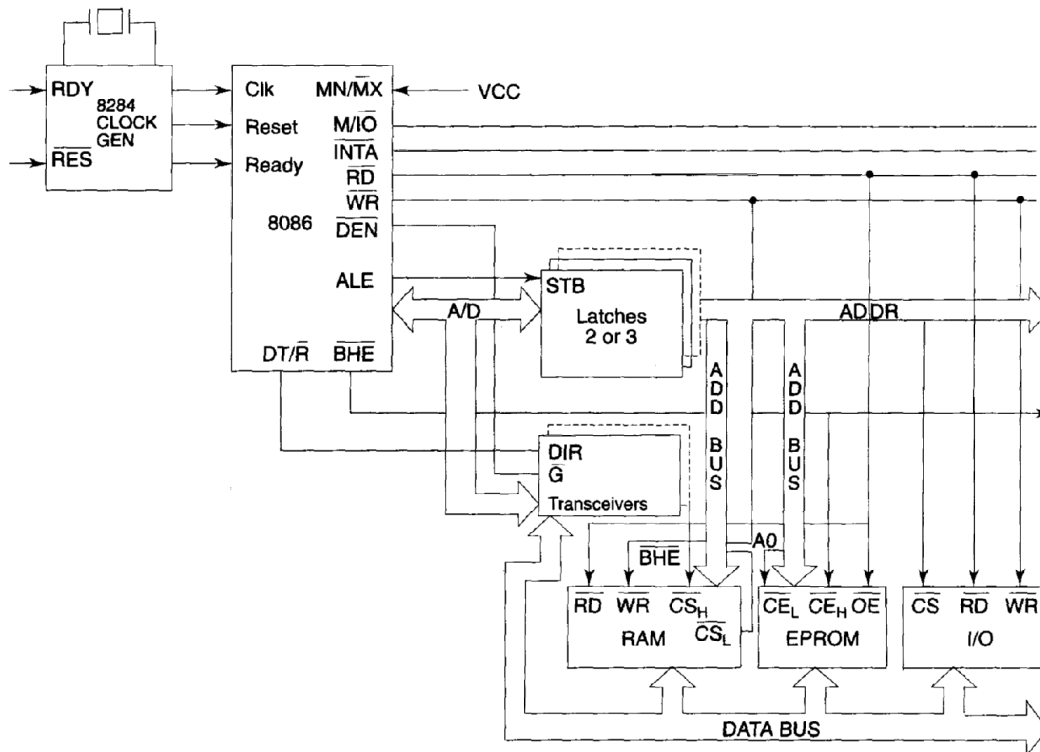


Fig.1.8 Minimum Mode 8086 System

The working of the minimum mode configuration system can be better described in terms of the timing diagrams rather than qualitatively describing the operations. The opcode fetch and read cycles are similar. Hence the timing diagram can be categorized in two parts, the first is the timing diagram for read cycle and the second is the timing diagram for write cycle.

The read cycle begins in T_1 with the assertion of the address latch enable (ALE) signal and also M/I/O signal. During the negative going edge of this signal, the valid address is latched on the local bus. The BHE and A0 signals address low, high or both bytes. From T_1 to T_4 , the $\overline{M/\overline{IO}}$ signal indicate a memory or I/O operation. At T_2 , the address is removed from the local bus and is sent to the output. The bus is then tristated. The read (\overline{RD}) control signal is also activated in T_2 . The read (\overline{RD}) signal causes the addressed device to enable its data bus drivers. After \overline{RD} goes low, the valid data is available on the data bus. The addressed device will drive the READY line high. When the processor returns the read signal to high level, the addressed device will again tristate its bus drivers.

A write cycle also begins with the assertion of ALE and the emission of the address. The $\overline{M/\text{IO}}$ signal is again asserted to indicate a memory or I/O operation. In T_2 , after sending the address in T_1 , the processor sends the data to be written to the addressed location. The data remains on the bus until middle of T_4 state. The \overline{WR} becomes active at the beginning of T_2 (unlike \overline{RD} is somewhat delayed in T_2 to provide time for floating).

The \overline{BHE} and A_0 signals are used to select the proper byte or bytes of memory or I/O word to be read or written.

The $\overline{M/\text{IO}}$, \overline{RD} and \overline{WR} signals indicate the types of data transfer as specified in Table 1.5.

$\overline{M/\text{IO}}$	\overline{RD}	\overline{WR}	Indications
0	0	1	I/O Read
0	1	0	I/O Write
1	0	1	Memory Read
1	1	0	Memory Write

Figure 1.9(a) shows the read cycle while the Fig. 1.9(b) shows the write cycle.

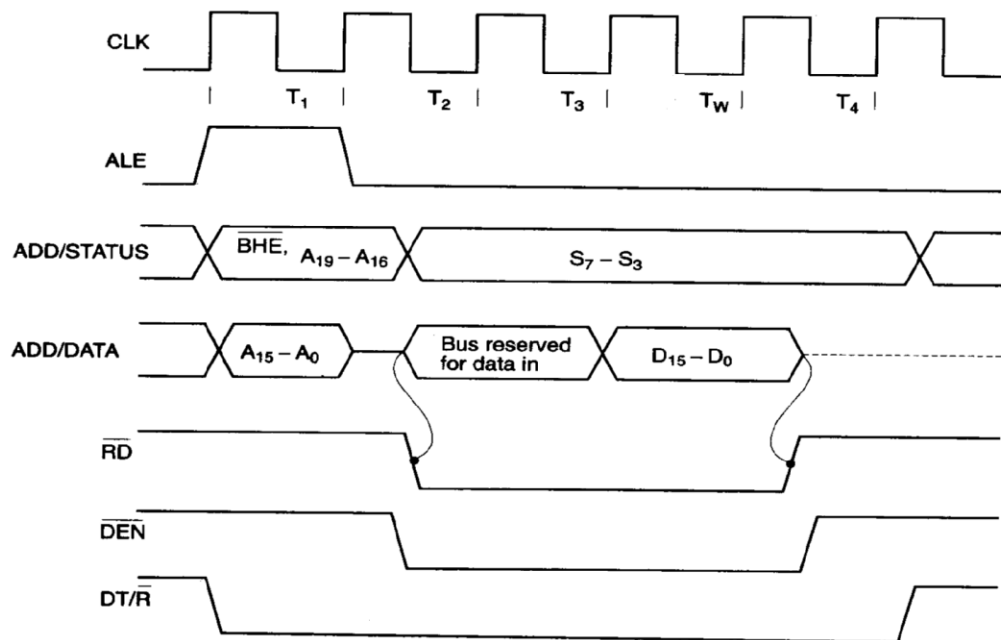


Fig.1.9(a) Read Cycle Timing Diagram for Minimum Mode

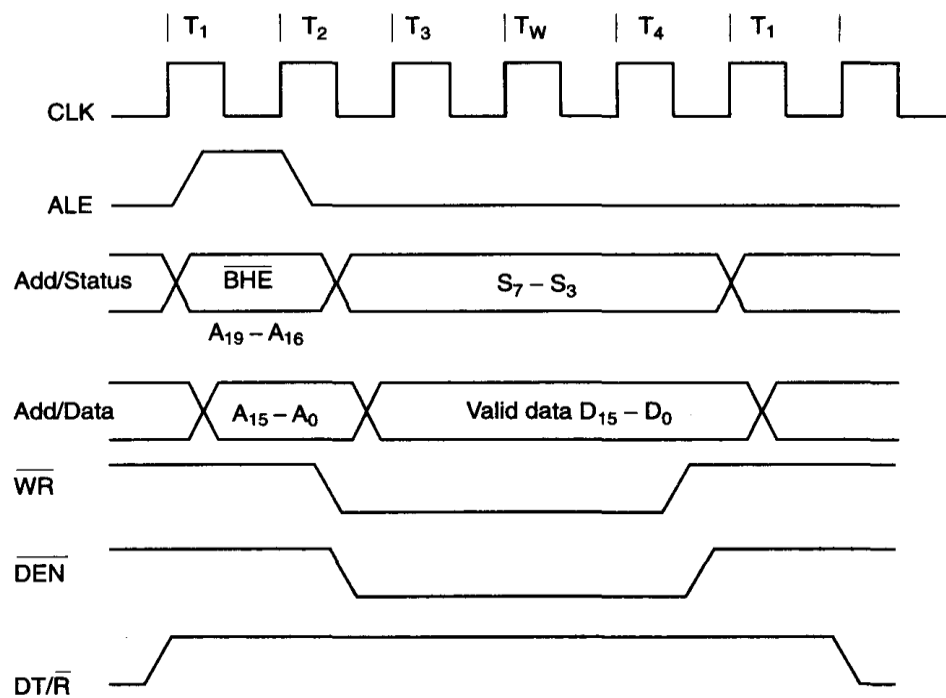


Fig.1.9(b) Write Cycle Timing Diagram for Minimum Operation

MAXIMUM MODE 8086 SYSTEM AND TIMINGS

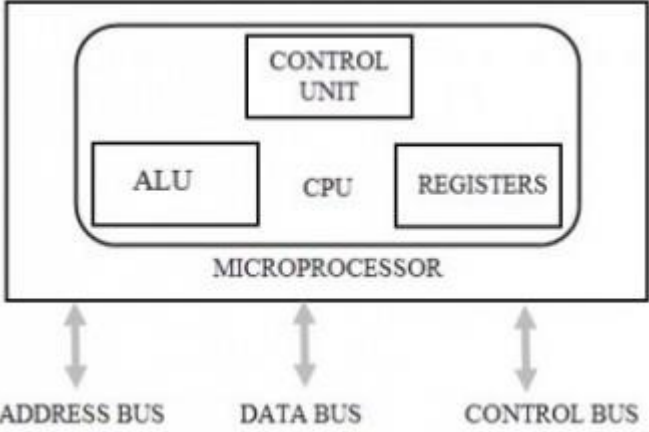
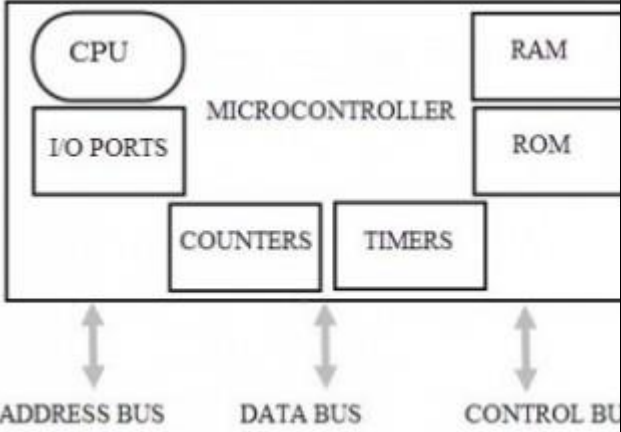
In the maximum mode, the 8086 is operated by strapping the MN/ $\overline{\text{MX}}$ pin to ground. In this mode, the processor derives the status signals $\overline{\text{S}}_2$, $\overline{\text{S}}_1$ and $\overline{\text{S}}_0$. Another chip called bus controller derives the control signals using this status information. In the maximum mode, there may be more than one microprocessor in the system configuration. The other components in the system are the same as in the minimum mode system.

The basic functions of the bus controller chip 1C8288, is to derive control signals like $\overline{\text{RD}}$ and $\overline{\text{WR}}$ (for memory and I/O devices), $\overline{\text{DEN}}$, $\text{DT}/\overline{\text{R}}$, ALE, etc. using the information made available by the processor on the status lines. The bus controller chip has input lines and $\overline{\text{S}}_2$, $\overline{\text{S}}_1$ and $\overline{\text{S}}_0$ CLK. These inputs to 8288 are driven by the CPU. It derives the outputs ALE, $\overline{\text{DEN}}$, $\text{DT}/\overline{\text{R}}$, $\overline{\text{MRDC}}$, $\overline{\text{MWTC}}$, $\overline{\text{AMWC}}$, $\overline{\text{IORC}}$, $\overline{\text{IOWC}}$ and $\overline{\text{AIOWC}}$. The $\overline{\text{AEN}}$, IOB and CEN pins are specially useful for multiprocessor systems. $\overline{\text{AEN}}$ and IOB are generally grounded. CEN pin is usually tied to +5V. The significance of the MCE/ $\overline{\text{PDEN}}$ output depends upon the status of the IOB pin. If IOB is grounded, it acts as master cascade enable to control cascaded

8259A, else it acts as peripheral data enable used in the multiple bus configurations. $\overline{\text{INTA}}$ pin is used to issue two interrupt acknowledge pulses to the interrupt controller or to an interrupting device.

$\overline{\text{IORC}}$, $\overline{\text{IOWC}}$ are I/O read command and I/O write command signals respectively. These signals enable an IO interface to read or write the data from or to the addressed port. The $\overline{\text{MRDC}}$, $\overline{\text{MWTC}}$ are memory read command and memory write command signals respectively and may be used as memory read and write signals. All these command signals instruct the memory to accept or send data from or to the bus. For both of these write command signals, the advanced signals namely $\overline{\text{AMWC}}$ and $\overline{\text{AIOWC}}$ are available. They also serve the same purpose, but are activated one clock cycle earlier than the $\overline{\text{IOWC}}$ and $\overline{\text{MWTC}}$ signals, respectively. The maximum mode system is shown in Fig. 1.10.

The maximum mode system timing diagrams are also divided in two portions as read (input) and write (output) timing diagrams. The address/data and address/status timings are similar to the minimum mode. ALE is asserted in T_1 , just like minimum mode. The only difference lies in the status signals used and the available control and advanced command signals.

MICROPROCESSOR	MICROCONTROLLER
	
<p>Microprocessor assimilates the function of a central processing unit (CPU) on to a single integrated circuit (IC).</p>	<p>Microcontroller can be considered as a small computer which has a processor and some other components in order to make it a computer.</p>
<p>Microprocessors are mainly used in designing general purpose systems from small to large and complex systems like super computers.</p>	<p>Microcontrollers are used in automatically controlled devices.</p>
<p>Microprocessors are basic components of personal computers.</p>	<p>Microcontrollers are generally used in embedded systems</p>
<p>Computational capacity of microprocessor is very high. Hence can perform complex tasks.</p>	<p>Less computational capacity when compared to microprocessors. Usually used for simpler tasks.</p>
<p>A microprocessor based system can perform numerous tasks.</p>	<p>A microcontroller based system can perform single or very few tasks.</p>
<p>Microprocessors have integrated Math Coprocessor. Complex mathematical calculations which involve floating point can be performed with great ease.</p>	<p>Microcontrollers do not have math coprocessors. They use software to perform floating point calculations which slows down the device.</p>
<p>The main task of microprocessor is to perform the instruction cycle repeatedly. This includes fetch, decode and execute.</p>	<p>In addition to performing the tasks of fetch, decode and execute, a microcontroller also controls its environment based on the output of the instruction cycle.</p>

MICROPROCESSOR	MICROCONTROLLER
In order to build or design a system (computer), a microprocessor has to be connected externally to some other components like Memory (RAM and ROM) and Input / Output ports.	The IC of a microcontroller has memory (both RAM and ROM) integrated on it along with some other components like I / O devices and timers.
The overall cost of a system built using a microprocessor is high. This is because of the requirement of external components.	Cost of a system built using a microcontroller is less as all the components are readily available.
Generally power consumption and dissipation is high because of the external devices. Hence it requires external cooling system.	Power consumption is less.
The clock frequency is very high usually in the order of Giga Hertz.	Clock frequency is less usually in the order of Mega Hertz.
Instruction throughput is given higher priority than interrupt latency.	In contrast, microcontrollers are designed to optimize interrupt latency.
Have few bit manipulation instructions	Bit manipulation is powerful and widely used feature in microcontrollers. They have numerous bit manipulation instructions.
Generally microprocessors are not used in real time systems as they are severely dependent on several other components.	Microcontrollers are used to handle real time tasks as they are single programmed, self sufficient and task oriented devices.

1.10 Learning Outcomes:

1. Know:

- To describe the features, software architecture (Programmer's model / Register set) and Hardware architecture (Block Diagram) of Intel 8086 / 8088 processor. (
- To Use of memory segmentation with its pros and cons.
- To Describe the interrupt structure and how 8086 respond to interrupt

2. Comprehend:

- To distinguish between 8086 and 8088 processor

3. Apply, analyze and synthesize:

e) To Draw the minimum mode CPU configuration and memory cycles.

f) To Draw the maximum mode CPU configuration for 8086 and the functions of Intel 8288 bus controller.

1.11. Viva Questions

Q.1 What is segmentation?

Ans: It is the process in which the main memory of computer is divided into different **segments** and each **segment** has its own base address. * **Segmentation** is used to increase the execution speed of computer system so that processor can able to fetch and execute the data from memory easily and fastly.

Q.2 Define Memory mapped I/O?

Ans: Instead of a memory register, if an output device is connected at the address, the accumulator contents will be transferred to the output device. This is called memory mapped I/O

Q.3 What is the Maximum clock frequency in 8086?

Ans: 5 Mhz is the Maximum clock frequency in 8086.

Q.4 What is meant by Maskable interrupts?

Ans:An interrupt that can be turned off by the programmer is known as Maskable interrupt.

Q.5 What is Non-Maskable interrupts?

Ans:An interrupt which can be never be turned off (ie. disabled) is known as Non-Maskable interrupt

Q.6 What are the different functional units in 8086?

Ans:Bus Interface Unit and Execution unit, are the two different functional units in 8086.

Q.7 What are the various segment registers in 8086?

Ans:Code, Data, Stack, Extra Segment registers in 8086.

Q.8 What does EU do?

Ans:Execution Unit receives program instruction codes and data from BIU, executes these instructions and store the result in general registers.

Q.9 Which Stack is used in 8086?

Ans:FIFO (First In First Out) stack is used in 8086.In this type of Stack the first stored information is retrieved first.

Q.10 What are the flags in 8086?

Ans:In 8086 Carry flag, Parity flag, Auxiliary carry flag, Zero flag, Overflow flag, Trace flag, Interrupt flag, Direction flag, and Sign flag.

1.12. Multiple Choice Questions.

1. 8088 microprocessor differs with 8086 microprocessor in
2. i) Data width on the output ii) Address capability
3. iii)Support of coprocessor iv)Support of MAX / MIN mode

4. In 8086 microprocessor one of the following statements is not true.
 - a) Coprocessor is interfaced in MAX mode
 - b) Coprocessor is interfaced in MIN mode
 - c) I/O can be interfaced in MAX / MIN mode
 - d) Support Pipelining

5. Which of the processor listed below have 16-bit internal data registers?
 - a) 8086
 - b) 8088
 - c) 80286
 - d) 80386

6. A microprocessor is a chip integrating all the functions of a CPU of a computer.
A. multiple B. single C. double D. triple
7. The first digital electronic computer was built in the year
A. 1950 B. 1960 C. 1940 D. 1930
8. In 1960's texas institute invented
A. integrated circuits B. microprocessor C. vacuum tubes D. transistors
9. The microprocessor can read/write 16 bit data from or to
A. memory B. I/O device C. processor D. register

10. The 16 bit flag of 8086 microprocessor is responsible to indicate
A. the condition of result of ALU operation B. the condition of memory C. the result of addition D. the result of subtraction

11. The BIU contains FIFO register of size bytes
A. 8 B. 6 C. 4 D. 12

12. The 1 MB byte of memory can be divided into segment
A. 1 Kbyte B. 64 Kbyte C. 33 Kbyte D. 34 Kbyte
13. The RD, WR, M/IO is the heart of control for a mode

A. minimum B. maximum C. compatibility mode D. control mode

14. If MN/MX is low the 8086 operates in mode

A. Minimum B. Maximum C. both (A) and (B) D. medium

15. Primary function of memory interfacing is that the should be able to read from and write into register

A. multiprocessor B. microprocessor C. dual Processor D. coprocessor

16. In which year, 8086 was introduced?

A. 1978 B. 1979 C. 1977 D. 1981

17. ALE stands for

A. address latch enable B. address level enable C. address leak enable
D. address leak extension

18. The First Microprocessor was .

A. Intel 4004 B. 8080 C. 8085 D. 4008

1.13 Short questions

Q.1 Differentiate between BIU & EU of 8086 microprocessor.

Q.2 What are the advantages of memory segmentation?

Q.3 Explain with example how 20-bit physical address is formed in 8086.

Q.5 Explain the function of following pins:

a) $\overline{\text{Lock}}$ b) $\overline{\text{BHE}}$ c) NMI & INTR

d) HOLD & HLDA e) RESET

Q.4 How IP helps in producing physical address?

1.14. Long Questions:

Q.1. Write short note on 8288 Bus Controller.

Q.2 What is segmented memory? Enumerate the advantages of segmented memory with reference to the 8086 microprocessor.

Q.3 Explain suitable diagram architecture of Intel 8086 processor.

Q.4 Explain difference between Intel 8086 & 8088 processor.

Q.5 Draw and explain timing diagram for write operation in minimum mode of 8086.

Q.6 Draw and Explain Minimum mode CPU configuration of 8086 with neat diagram.

Q.7 Draw and Explain Maximum mode CPU configuration of 8086 with neat diagram.

1.15. References

1. Microprocessors and Interfacing ,Douglas V Hall,T ata Mc Graw Hill
2. http://www.telnet7.net/articles/not_mine/8086_achitecture.htm

3. The 8086/8088 Family, John Uffenbuck, Pearson Media, LPE
4. [www. nptel.ac.in/courses/106108100/](http://www.nptel.ac.in/courses/106108100/)
5. [www. google.com](http://www.google.com)

Self-Assessment

- Q.1 What is the function of flag register? Illustrate your answer with suitable example.
- Q.2 What are the default and specified segment registers?
- Q.3 Discuss in detail the segmentation of memory in 8086.
- Q.4 How SP helps in producing physical address?
- Q.5 Explain the functions of following pins:
a) LOCK b) TEST c) BHE'

Self-Evaluation

Name of Student		
Class		
Roll No.		
Subject		
Module No.		
S.No		Tick Your choice
1.	Do you understand the architecture of 8086	<input type="radio"/> Yes <input type="radio"/> No
2.	Do you understand the functions of pins	<input type="radio"/> Yes <input type="radio"/> No
3.	Do you understand the operating modes of 8086	<input type="radio"/> Yes <input type="radio"/> No
4.	Do you able to expalin memory segmentation?	<input type="radio"/> Yes <input type="radio"/> No
5.	Do you understand module ?	<input type="radio"/> Yes, Completely. <input type="radio"/> Partialy. <input type="radio"/> No, Not at all.

Module: 2

Instruction set and Assembly Language Programming

2.1. Motivation:

At the completion of the subject, students should be able to: study and write Assembly Language Programming (ALP)

2.2. Syllabus:

Lecture	Content	Duration	Self Study
9 & 10	8086 Instruction Set & Programming: Instruction Set of 8086	2 Lecture	2 hours
11	Instruction Set of 8086	1 Lecture	2 hours
12	Addressing Modes	1 Lecture	2 hours
13	Assembly Language Programming	1 Lecture	2 hours
14	Mixed Language Programming with C Language and Assembly Language	1 Lecture	2 hours

2.3. Weightage: 15 Marks

2.4. Learning Objectives: Students should be able to-

1. Explain all the instructions with examples, formats and their use in the programming. (U)
2. Write assembly language program for 8086 using instruction set, assembler directives and addressing modes etc.(R)
3. Assess assembly language programs using TASM/MASM which are run on 8086 microprocessor system (E)
4. To Compare between Macros and procedure and Specify when to use macro and procedure .(E)
5. To define macros as and when required to increase readability, productivity of program. (R)
6. Experiment how to use C/C++ features in assembly language to make program for interactive and user friendly i.e . mixed mode programming (AN)

2.5. Theoretical Background:

The 8086 instruction set includes equivalents of the 8085 instructions plus many new ones. The new instructions contain operations such as signed/unsigned multiplication and division, bit manipulation instructions, string instructions and interrupt instructions.

The 8086 has approximately 117 different instructions with about 300 opcodes. The 8086 instruction set contains no operand, single operand, and two operand instructions. Except for string instructions which involve array operations, the 8086 instructions do not permit memory to memory operations.

2.6. Abbreviations:

4.7. Formulae:Nil

4.8. Key Definitions:

- Addressing Mode:- An instruction acts on any number of operands. The way an instruction accesses its operands is called its Addressing modes.
- Assembly Level Language: - *Assembly Language* is a low level programming language using the human readable instructions of the CPU.
- Addressing Mode:- The way an instruction accesses its operands is called its Addressing modes.

2.9. Course Content:

Lecture: 9 & 10

8086 Instruction Set & Programming: Instruction Set of 8086

Learning Objective: In this lecture student will able to learn about **8086 Instruction Set & Programming:**

2.9.1 Instruction Set of 8086-

The 8086 microprocessor supports 8 types of instructions –

- Data Transfer Instructions
- Arithmetic Instructions
- Bit Manipulation Instructions
- String Instructions
- Program Execution Transfer Instructions (Branch & Loop Instructions)
- Processor Control Instructions
- Iteration Control Instructions
- Interrupt Instructions

Data Transfer Instructions

These instructions are used to transfer the data from the source operand to the destination operand. Following are the list of instructions under this group –

Instruction to transfer a word

- **MOV** – Used to copy the byte or word from the provided source to the provided destination.
- **PPUSH** – Used to put a word at the top of the stack.
- **POP** – Used to get a word from the top of the stack to the provided location.
- **PUSHA** – Used to put all the registers into the stack.
- **POPA** – Used to get words from the stack to all registers.
- **XCHG** – Used to exchange the data from two locations.
- **XLAT** – Used to translate a byte in AL using a table in the memory.

Instructions for input and output port transfer

- **IN** – Used to read a byte or word from the provided port to the accumulator.
- **OUT** – Used to send out a byte or word from the accumulator to the provided port.

Instructions to transfer the address

- **LEA** – Used to load the address of operand into the provided register.
- **LDS** – Used to load DS register and other provided register from the memory
- **LES** – Used to load ES register and other provided register from the memory.

Instructions to transfer flag registers

- **LAHF** – Used to load AH with the low byte of the flag register.
- **SAHF** – Used to store AH register to low byte of the flag register.
- **PUSHF** – Used to copy the flag register at the top of the stack.
- **POPF** – Used to copy a word at the top of the stack to the flag register.

Arithmetic Instructions

These instructions are used to perform arithmetic operations like addition, subtraction, multiplication, division, etc.

Following is the list of instructions under this group –

Instructions to perform addition

- **ADD** – Used to add the provided byte to byte/word to word.
- **ADC** – Used to add with carry.
- **INC** – Used to increment the provided byte/word by 1.

- **AAA** – Used to adjust ASCII after addition.
- **DAA** – Used to adjust the decimal after the addition/subtraction operation.

Instructions to perform subtraction

- **SUB** – Used to subtract the byte from byte/word from word.
- **SBB** – Used to perform subtraction with borrow.
- **DEC** – Used to decrement the provided byte/word by 1.
- **NPG** – Used to negate each bit of the provided byte/word and add 1/2's complement.
- **CMP** – Used to compare 2 provided byte/word.
- **AAS** – Used to adjust ASCII codes after subtraction.
- **DAS** – Used to adjust decimal after subtraction.

Instruction to perform multiplication

- **MUL** – Used to multiply unsigned byte by byte/word by word.
- **IMUL** – Used to multiply signed byte by byte/word by word.
- **AAM** – Used to adjust ASCII codes after multiplication.

Instructions to perform division

- **DIV** – Used to divide the unsigned word by byte or unsigned double word by word.
- **IDIV** – Used to divide the signed word by byte or signed double word by word.
- **AAD** – Used to adjust ASCII codes after division.
- **CBW** – Used to fill the upper byte of the word with the copies of sign bit of the lower byte.
- **CWD** – Used to fill the upper word of the double word with the sign bit of the lower word.

Bit Manipulation Instructions

These instructions are used to perform operations where data bits are involved, i.e. operations like logical, shift, etc.

Following is the list of instructions under this group –

Instructions to perform logical operation

- **NOT** – Used to invert each bit of a byte or word.
- **AND** – Used for adding each bit in a byte/word with the corresponding bit in another byte/word.
- **OR** – Used to multiply each bit in a byte/word with the corresponding bit in another

byte/word.

- **XOR** – Used to perform Exclusive-OR operation over each bit in a byte/word with the corresponding bit in another byte/word.
- **TEST** – Used to add operands to update flags, without affecting operands.

Instructions to perform shift operations

- **SHL/SAL** – Used to shift bits of a byte/word towards left and put zero(S) in LSBs.
- **SHR** – Used to shift bits of a byte/word towards the right and put zero(S) in MSBs.
- **SAR** – Used to shift bits of a byte/word towards the right and copy the old MSB into the new MSB.

Instructions to perform rotate operations

- **ROL** – Used to rotate bits of byte/word towards the left, i.e. MSB to LSB and to Carry Flag [CF].
- **ROR** – Used to rotate bits of byte/word towards the right, i.e. LSB to MSB and to Carry Flag [CF].
- **RCR** – Used to rotate bits of byte/word towards the right, i.e. LSB to CF and CF to MSB.
- **RCL** – Used to rotate bits of byte/word towards the left, i.e. MSB to CF and CF to LSB.

String Instructions

String is a group of bytes/words and their memory is always allocated in a sequential order.

Following is the list of instructions under this group –

- **REP** – Used to repeat the given instruction till $CX \neq 0$.
- **REPE/REPZ** – Used to repeat the given instruction until $CX = 0$ or zero flag $ZF = 1$.
- **REPNE/REPNZ** – Used to repeat the given instruction until $CX = 0$ or zero flag $ZF = 1$.
- **MOVS/MOVSMB/MOVSMB** – Used to move the byte/word from one string to another.
- **COMS/COMPSMB/COMPSW** – Used to compare two string bytes/words.
- **INS/INSM/INSW** – Used as an input string/byte/word from the I/O port to the provided memory location.
- **OUTS/OUTSM/OUTSW** – Used as an output string/byte/word from the provided memory location to the I/O port.
- **SCAS/SCASMB/SCASW** – Used to scan a string and compare its byte with a byte in AL or string word with a word in AX.

- **LODS/LODSB/LODSW** – Used to store the string byte into AL or string word into AX.

Lecture: 11

Learning Objective: In this lecture student will able to learn about **8086 Instruction Set & Programming**

2.9.2 8086 Instruction Set

Program Execution Transfer Instructions (Branch and Loop Instructions)

These instructions are used to transfer/branch the instructions during an execution. It includes the following instructions –

Instructions to transfer the instruction during an execution without any condition –

- **CALL** – Used to call a procedure and save their return address to the stack.
- **RET** – Used to return from the procedure to the main program.
- **JMP** – Used to jump to the provided address to proceed to the next instruction.

Instructions to transfer the instruction during an execution with some conditions –

- **JA/JNBE** – Used to jump if above/not below/equal instruction satisfies.
- **JAE/JNB** – Used to jump if above/not below instruction satisfies.
- **JBE/JNA** – Used to jump if below/equal/ not above instruction satisfies.
- **JC** – Used to jump if carry flag CF = 1
- **JE/JZ** – Used to jump if equal/zero flag ZF = 1
- **JG/JNLE** – Used to jump if greater/not less than/equal instruction satisfies.
- **JGE/JNL** – Used to jump if greater than/equal/not less than instruction satisfies.
- **JL/JNGE** – Used to jump if less than/not greater than/equal instruction satisfies.
- **JLE/JNG** – Used to jump if less than/equal/if not greater than instruction satisfies.
- **JNC** – Used to jump if no carry flag (CF = 0)
- **JNE/JNZ** – Used to jump if not equal/zero flag ZF = 0
- **JNO** – Used to jump if no overflow flag OF = 0
- **JNP/JPO** – Used to jump if not parity/parity odd PF = 0
- **JNS** – Used to jump if not sign SF = 0
- **JO** – Used to jump if overflow flag OF = 1
- **JP/JPE** – Used to jump if parity/parity even PF = 1
- **JS** – Used to jump if sign flag SF = 1

Processor Control Instructions

These instructions are used to control the processor action by setting/resetting the flag values.

Following are the instructions under this group –

- **STC** – Used to set carry flag CF to 1
- **CLC** – Used to clear/reset carry flag CF to 0
- **CMC** – Used to put complement at the state of carry flag CF.
- **STD** – Used to set the direction flag DF to 1
- **CLD** – Used to clear/reset the direction flag DF to 0
- **STI** – Used to set the interrupt enable flag to 1, i.e., enable INTR input.
- **CLI** – Used to clear the interrupt enable flag to 0, i.e., disable INTR input.

Iteration Control Instructions

These instructions are used to execute the given instructions for number of times. Following is the list of instructions under this group –

- **LOOP** – Used to loop a group of instructions until the condition satisfies, i.e., $CX = 0$
- **LOOPE/LOOPZ** – Used to loop a group of instructions till it satisfies $ZF = 1$ & $CX = 0$
- **LOOPNE/LOOPNZ** – Used to loop a group of instructions till it satisfies $ZF = 0$ & $CX = 0$
- **JCXZ** – Used to jump to the provided address if $CX = 0$

Interrupt Instructions

These instructions are used to call the interrupt during program execution.

- **INT** – Used to interrupt the program during execution and calling service specified.
- **INTO** – Used to interrupt the program during execution if $OF = 1$
- **IRET** – Used to return from interrupt service to the main program

Let's check the take away from this lecture

1) Which of the following is an illegal 8086 instruction?

- (a) MOV 20h, BX (b) INC AL (c) AND BX, BX (d) ADD AX, 30h

2. Which of the following is an illegal 8086 instruction

- (a) MOV AX, [BX] (b) INC [BX] (c) ADD [AX], [BX] (d) ADD AX, [CX]

3..Branch prediction is used in the context of

- (a) pipelining (b) program loops (c) cache memory (d) ALU operation

4. A conditional jump instruction

- (a) always cause a transfer of control

- (b) always involves the use of the status register
- (c) always modifies the program counter
- (d) always involves testing the Zero flag

5. The advantage of memory mapped I/O over I/O mapped I/O is,

- (a) Many instructions supporting memory mapped I/O
- (b) Faster
- (c) Require a bigger address decoder
- (d) All the above

L12. Exercise:

Q.1 Discuss all types of jump instructions used in 8086 microprocessor

Q. 2 Explain in detail the difference between near CALL and far CALL

Questions/problems for practice:

Q. 3 Explain the following instructions:

- a. XLAT
- b. RCL

Q. 4 Explain the use of PUSH and POP instructions in 8086

Learning from the lecture '8086 Instruction Set':

Student will be able to apply 8086 Instruction set.

Lecture: 12

Learning Objective: In this lecture student will be able to learn about **Addressing Modes of 8086 Microprocessor**

2.9.3 8086 Addressing Modes

- a) Register Addressing Mode
- b) Immediate Addressing Mode
- c) Direct addressing Mode
- d) Register indirect Addressing Mode
- e) Register Relative Addressing Mode
- f) Base-plus-index Addressing Mode
- g) Base Index Relative Addressing Mode
- h) Implied Addressing Mode

(a) Register Addressing Mode:-

In this mode the source operand as well as destination operand both are to be contained in the 8086 register.

e.g. MOV DX, CX... Here content of CX is moved to DX register.

MOV CL, DL....Here content of DL is moved to CL register

- 8-bit registers : AH, AL, BH, BL, CH, CL, DH and DL.
- 16-bit registers: AX, BX, CX, DX, SP, BP, SI, and DI.
- *never* mix an 8-bit with a 16-bit register or vice versa as a source and destination.

(b) Immediate Addressing Mode:-

- In this mode destination is always constant value & it transfers immediately into the destination and destination is always register.

e.g. MOV CL, 03H...Here value 03h move to the CL register.

MOV DX, 0502H... Here 16-bit value 0502h moves to the DX register.

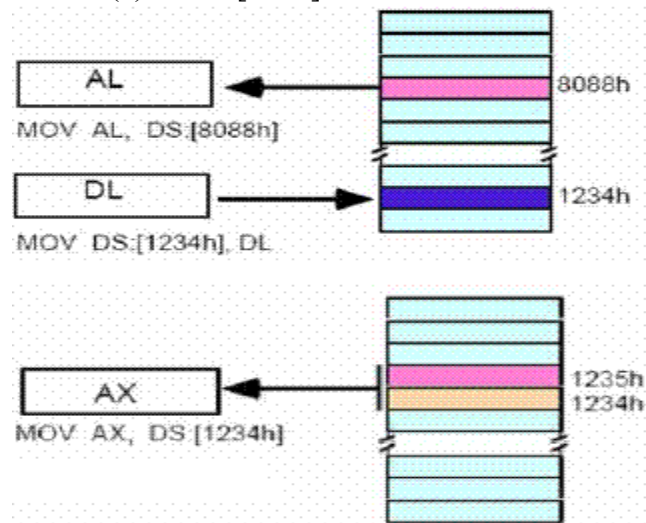
- Term immediate implies that data immediately follow the hexadecimal opcode in the memory.
- Immediate data are constant data
- The letter H appends hexadecimal data.
- Decimal data are represented as without any special codes or adjustments.
- An example is the 100 decimal in the MOV AL, 100 instructions.

(C) Direct Addressing Mode:-

- In this addressing mode source or destination, any one is always memory location and always direct address is specifying in the instruction itself.

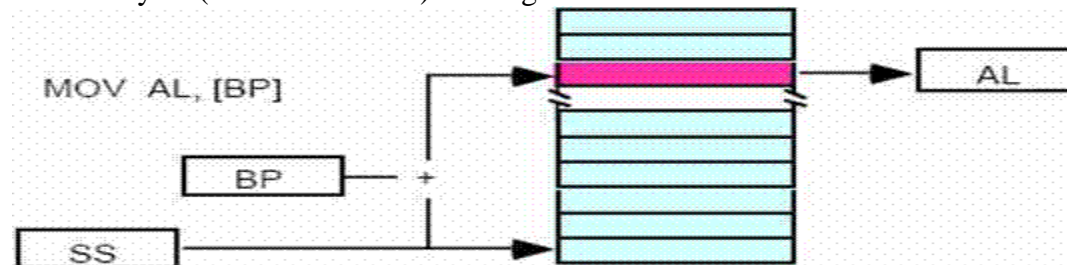
e.g. (1) MOV AL, [8088].....Here content of 8088h (i.e. DS*10 + 8088H) memory location moves to the register AL.

(2) MOV [4400], AH....Here content of AH register move to the address 4400H.

**(D) Register Indirect Addressing Mode:-**

- In this Addressing Mode direct address is not specifying in the instruction instead of that address specifying with the help of register indirectly.

e.g. MOV AL,[BP] instruction copies the word-sized data from the data segment offset address indexed by SI (i.e. SS*10h +BP) into register AL.

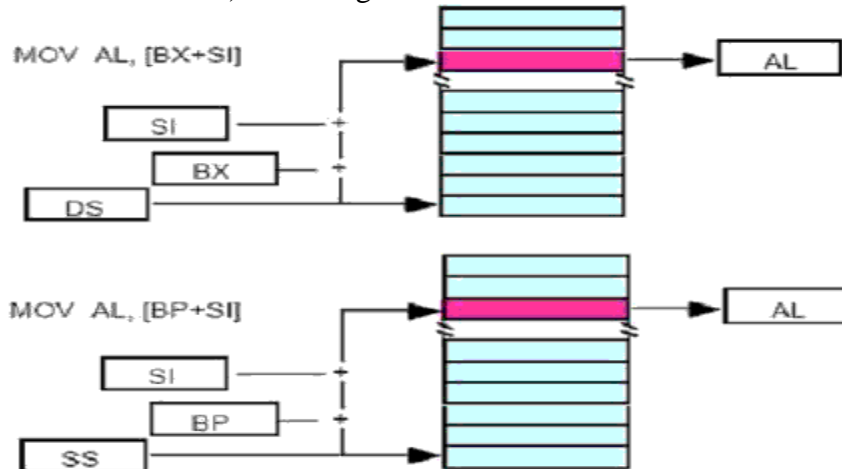


(E) Base-plus-index Addressing Mode:-

- In this Addressing Mode it will Transfers a byte or word between a register and the memory location addressed by a base register (BP or BX) plus an index register (DI or SI).

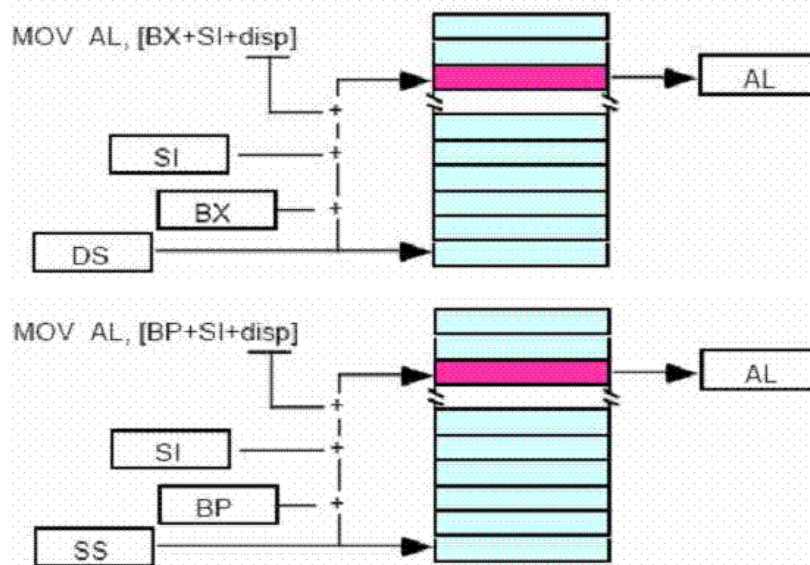
e.g. (1) `MOV [BX+DI],CL` ...Here instruction copies the byte-sized contents of register CL into the data Segment memory location addressed by BX+DI (i.e. $ES*10 + DI + BX$).

(2) `MOV AL,[BX+SI]`.... Here instruction copies the byte-sized contents pointed by BX+SI (i.e. $DS*10 + SI + BX$) in AL register.

**(F) Base Index Relative Addressing Mode:-**

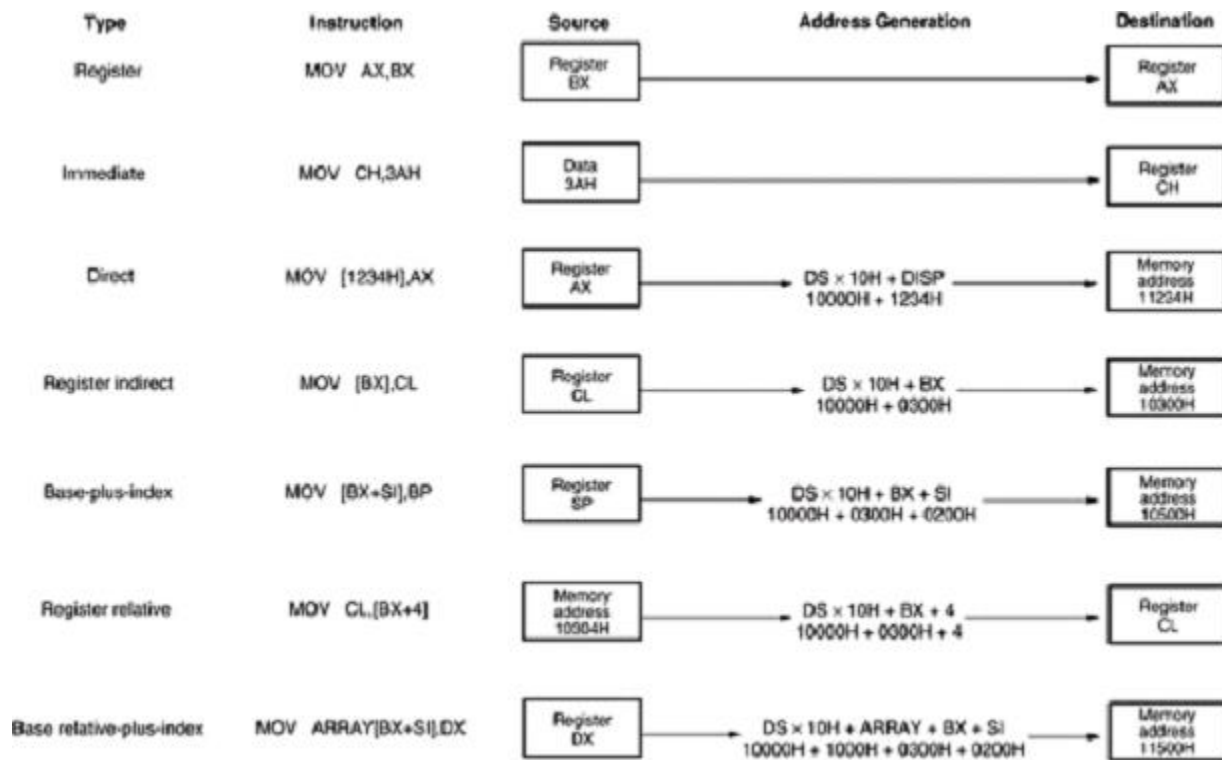
- In this Addressing Mode it will Transfers a byte or word between a register and the memory location addressed by a base register (BP or BX) plus an index register (DI or SI) plus displacement.

e.g. `MOV AL, [BX+SI+25h]`.... Here instruction copies the byte-sized contents pointed by $BX+SI+25h$ (i.e. $DS*10 + SI + BX + 25h$) into AL register.



(G) Implied Addressing

- In this Addressing Mode instruction do not have any operand.
e.g. CLC...Here it will clear the carry flag (i.e. CY = 0).



The following table summarizes all addressing modes used by the 8086 processor.

Addressing Mode	Example	Source operand		
		Assuming: DS = 1000H, BX = 0200H, SI = 0300H		
		Type	Address Generation	Address
Register	MOV AX,BX	Register	-	-
Immediate	MOV AX, 0F7H	Immed.	-	-
Direct	MOV AX,[1234H]	Mem.	$DS \times 10H + 1234H$	11234H
Register-Indirect	MOV AX,[BX]	Mem.	$DS \times 10H + 0200H$	10200H
Based	MOV AX,[BX+06]	Mem.	$DS \times 10H + 0200H + 0006H$	10206H
Indexed	MOV AX,[SI+06]	Mem.	$DS \times 10H + 0300H + 0006H$	10306H
Based-Indexed	MOV AX,[BX+SI+06]	Mem.	$DS \times 10H + 0200H + 0300H + 0006H$	10506H

Let's check the take away from this lecture

1) Which of the following is an illegal 8086 instruction?

- (a) MOV 20h,BX (b) INC AL (c) AND BX,BX (d) ADD AX,30h

2. Which of the following is an illegal 8086 instruction

- (a) MOV AX,[BX] (b) INC [BX] (c) ADD [AX], [BX] (d) ADD AX, [CX]

3..Branch prediction is used in the context of

- (a) pipelining (b) program loops (c) cache memory (d) ALU operation

4. A conditional jump instruction

- (a) always cause a transfer of control
(b) always involves the use of the status register
(c) always modifies the program counter
(d) always involves testing the Zero flag

5. The advantage of memory mapped I/O over I/O mapped I/O is,

- (a) Many instructions supporting memory mapped I/O (b) Faster
(c) Require a bigger address decoder (d) All the above

L13. Exercise:

Q.1 What is addressing mode? Explain different modes supported by 8086 with examples.

Questions/problems for practice:

Q. 3 2 Explain memory organization concept of 8086.

.

Learning from the lecture ‘8086 Addressing Modes’:

Student will able to understand 8086 Addressing modes.

Lecture: 13 & 14

Learning Objective: In this lecture student will able to learn about **Assembly Language Programming**

2.9.4 Assembly Language Programming

Assembler Directives:-

This type of statements includes commands that are addressed to the assembler, such as:
Constant and variable definition.
Allocation of memory space and initialization of memory, and
Control of the assembly process

List of assembler directives

a. Data Allocation Directives

DB.....define byte

DW.....define word (2 bytes)

DD.....define double word (4 bytes)

DQ.....define quadword (8 bytes)

DT.....define tenbytes

EQU.....equate, assign numeric expression to a name

Examples:

db 100 dup (?) define 100 bytes, with no initial values for bytes

db "Hello" define 5 bytes, ASCII equivalent of "Hello".

maxint equ 32767

count equ 10 * 20 ; calculate a value (200)

b. Segment definition directives:

A segment can be defined in two ways:

Segment definition

The form which is used to define a segment is as follows:

Seg_name SEGMENT

.....

<*segment_statements*>

.....

Seg_name ENDS

Example

Code_Here SEGMENT

ASSUME CS:Code_Here

.....

.....

Code_Here ENDS

END

ENDS.....used to indicate the end of the segment.

END..... used to indicate the end of program.

PROC.....used to indicate the beginning of a procedure.

ENDP..... used to indicate the end of a program.

ENDM..... used to indicate the end of a program.

SEGMENT.....used to indicate the start of the segment.

TITLE.....used to indicate the title of the program.

EQUused to give a name to some value or to a symbol. Each time the assembler finds the name in the program, it will replace the name with the value or symbol you given to that name.

ASSUME..... associates a logical segment to processor segment.

e.g. Example:

ASSUME CS:CODE ;

This tells the assembler that the logical segment named CODE contains the instruction statements for the program and should be treated as a code segment.

ASSUME DS:DATA ;

This tells the assembler that for any instruction which refers to a data in the data segment, data will found in the logical segment DATA.

Write assembly language program for 8086 to reverse string of 10 characters.

	.data
	src_block db 01,02,03,04,05,06,07,08,09,0AH
	dest_block db 10dup(?)
	Count dw 0AH
	.code
	Mov ax,@data
	Mov ds,ax
	Mov es,ax
	Mov si,offset src_block
	Mov di,offset dest_block
	Mov cx,count
	cld
Again :	Rep movsb
	Xchg
	Mov di,offset dest_block
	Mov bh,0AH
Up :	Mov bl,[di]
	Mov ah,4Ch
	Int 21h
	end

Let's check the take away from this lecture

1)

L12. Exercise:

Q.1 WAP for 8086 in assembly language to check if string initialized in the data segment is Palindrome or not.

2. Q. 2 What do you mean by assembler directive? Explain any four in detail?

Questions/problems for practice:

Q. 3WAP to add 4 digit BCD number in AX to the similar in BX maintaining valid BCD result

Learning from the lecture 'Assembly Language Programming:

Student will able to write programs using Assembly Language Programming.

2.10 Learning Outcomes:

1. Know:

1. To Explain all the instructions with examples, formats and their use in the programming.
- (U)

2. Write assembly language program for 8086 using instruction set, assembler directives and addressing modes etc. **(R)**

2. Comprehend:

3. Assess assembly language programs using TASM/MASM which are run on 8086 microprocessor system **(E)**

4. To Compare between Macros and procedure and Specify when to use macro and procedure **(E)**

3. Apply, analyze and synthesize:

Experiment how to use C/C++ features in assembly language to make program for interactive and user friendly i.e . mixed mode programming **(AN)**

2.11. Viva questions

Q.1 Explain the following instructions:

- a. LAHF
- b. TEST
- c. AAD
- d. DAA

Q.2 Explain the usefulness of the following instructions in 8086

a. LOCK b. TEST c. XLAT d. LES

Q.3 Write the difference between the following instructions:

a. MOV CX, 437AH and MOV CX, [437AH]

Q. 4 With the help of an example describe the action performed by microprocessor 8086 for each of the following instructions

A. AAM B. CMPSB C. IMUL D. ROL

Q.5 Explain how 'Based Indexed' addressing mode of 8086 is useful for array element access.

2.12 MCQs

1. Instruction providing both segment base and offset address are called
 - A. below type .B. far type C. low type D. high type
2. The conditional branch instruction specify for branching
 - A. conditions B. instruction C. address D. memory
3. The BIU contains FIFO register of size 6 bytes called .
 - A. queue B. stack C. segment D. register
4. Which of the following is not an arithmetic instruction?

- A. INC (increment) B. CMP (compare) C. DEC (decrement) D. ROL (rotate left)
5. Which group of instructions do not affect the flags?
A. Arithmetic operations B. Logic operations C. Data transfer operations D. Branch operations
6. Which register is used as a default counter in case of string and loop instructions.
a) AX b) BX c) CX d) DX
7. Which is not part of execution unit?
a) ALU
b) Address conversion mechanism
c) Flag register
d) General purpose register
8. If segment address = 1005 H, offset address = 5555 H, then the physical address is_____.
a) 655A H
b) 155A5 H
c) 4550 H
d) 56555
9. _____ is the most important segment and it contains the actual assembly language instructions to be executed by the microprocessor.
a) Data segment
b) Code segment
c) Stack segment
d) Extra segment
10. Instruction Pointer (IP) contains offset address of _____ segment.
a) Data segment
b) Code segment
c) Stack segment
d) Extra segment

11. The index register is used to hold _____.
a) Segment memory
b) Offset memory
c) Offset address
d) Segment address
12. SI and DI registers is used to store the offset addresses of _____.
a) CS and DS or ES
b) DS and DS or ES
c) DS or ES and CS
d) DS and ES
13. The extension file that is must for a file to be accepted by the LINK as a valid object file is
A. .OBJ file
B. .EXE file
C. .MASM file
D. DEBUG file
14. Which of the following is an illegal 8086 instruction?
(a) MOV 20h, BX (b) INC AL
(c) AND BX, BX (d) ADD AX, 30h

2.13 Short questions

- Q.1 Write operations performed by the 8086 microprocessor CALL instruction.
Q.2 List all the string instructions of the 8086 microprocessor. Explain each of them in brief with a suitable example.
Q.3 Explain with the help of an example the string type of instructions of the 8086 microprocessor.
Q.4. Explain various addressing modes of 8086 microprocessor with examples.
Q.5 Expalin two indirect addressing modes of 8086 with examples.

2.13. Long Questions:

Q. 1. What is addressing mode? Explain different modes supported by 8086 with examples.

Q.2 Write a procedure to calculate the factorial of the N using recursive procedure.

Q.3 Differentiate between: procedure and macro.

Q.4 WAP for 8086 in assembly language to check if string initialized in the data segment is
Palindrome or not.

Q.5 Explain with example string instructions.

Q.6 WAP to add 4 digit BCD number in AX to the similar in BX maintaining valid BCD result.

Q.7 What do you mean by assembler directive? Explain any four in detail?

Q.8 Explain the following instructions:

1. XLAT
2. RCL
3. LAHF
4. TEST
5. SAR
6. DAS
7. AAD
8. DAA
9. CLD
10. MOVSB

2.15 References

- Microprocessors and Interfacing ,Douglas V Hall,T ata Mc Graw Hill
- http://www.telnet7.net/articles/not_mine/8086_achitecture.htm
- The 8086/8088 Family, John Uffenbuck, Pearson Media, LPE

Self-Assessment

Self-Evaluation

Name of Student		
Class		
Roll No.		
Subject		
Module No.		
S.No		Tick

		Your choice
6.	Do you understand the architecture of 8086	<input type="radio"/> Yes <input type="radio"/> No
7.	Do you understand the functions of pins	<input type="radio"/> Yes <input type="radio"/> No
8.	Do you understand the operating modes of 8086	<input type="radio"/> Yes <input type="radio"/> No
9.	Do you able to expalin memory segmentation?	<input type="radio"/> Yes <input type="radio"/> No
10.	Do you understand module ?	<input type="radio"/> Yes, Completely. <input type="radio"/> Partialy. <input type="radio"/> No, Not at all.

Module: 3

Memory and Peripheral Interfacing with 8086

3.1. Motivation:

At the completion of the subject, students should be able to:

- Motivation of this course is to provide the student knowledge of system design:
- The applications of Peripheral Controller namely 8255-PPI, 8253-PIT, 8259-PIC, and 8237-DMAC.
- System Design based on the Memory and Peripherals.

3.2. Syllabus:

Lecture	Content	Duration	Self Study
15	Memory Interfacing: SRAM, ROM	1 Lecture	2 hours
16	Memory Interfacing: SRAM, ROM: Minimum Mode	1 Lecture	2 hours
17	Memory Interfacing: SRAM, ROM: Maximum Mode	1 Lecture	2 hours
18	8259 PIC – Interrupt, Types of Interrupts, Interrupt Service Routine, Interrupt Vector Table	1 Lecture	2 hours
19	Block Diagram of 8259, Interfacing the 8259 in single and cascaded mode with 8086	1 Lecture	2 hours
20	8255 PPI - Block diagram, Command word format, Interfacing 8255 with 8086	1 Lecture	2 hours
21	Example: System Designing using Memory and Peripheral devices	1 Lecture	2 hours
22	Example: System Designing in Minimum mode	1 Lecture	2 hours
23	Example: System Designing in Maximum mode	1 Lecture	2 hours

3.3. Weightage: 30 Marks

3.4. Learning Objectives: Students should be able to-

1. Describe the working of 8259PIC. How to initialize and make 8259 Programmable to accept software and hardware interrupt requests in different operating modes .Also illustrate the working of cascade mode. (U)
2. To analyze the working of 8255 PPI. How it is used for 8 or 16 bit data transfer to /from peripheral devices and processor. (AN)
3. Illustrate the working of 8253 and show demonstration for different applications like Square wave generator, Interrupt on terminal, software trigger etc (A).
4. Explain the functionality of coprocessor and how it communicate with 8086 processor for execution of floating point operations. (E)
5. Distinguish between minimum and maximum mode and understand the concept of ODD and EVEN memory bank and how to interface SRAM and DRAM with 8086/8088 in minimum and maximum mode. (U)
6. Design microprocessor based system for given specification using minimum mode and maximum mode. (C)

3.5. Theoretical Background:

- The Intel 8254 Programmable Interval Timer (PIT) is a common medium scale integrated electronic device whose many and varied applications include integrating it as a featured component into simple microcomputer based systems. This device is also frequently used in Engineering and Engineering Technology classes when discussing computer hardware, microcomputer system architectures, or microprocessor interfacing. Often this device is utilized in the application portion of these classes or in student projects. The author presents an example of the programming of this device in MASM – an assembly language. This can be used as a boilerplate for describing the device's actions or in the use of this device in electronic or computer hardware laboratories or other applications.
- The Intel 8255A Programmable peripheral interface (PPI) is a general purpose programmable I/O device which is designed for use with all Intel and most other microprocessors. Its function is that of a general purposes I/O component to Interface peripheral equipment to the microcomputer system bush. The functional configuration of the 8255A is programmed by the systems software so that normally no external logic is necessary to interface peripheral devices or structures.
- It provides 24 I/O pins which may be individually programmed in 2 groups of 12 and used in 3 major modes of operation. It can interface any TTL-compatible I/O device to

microprocessor. The 8255 is a 40 pin integrated circuit (IC), designed to perform a variety of interface functions in a computer environment. The high performance and industry standard configuration of the 82C55A make it compatible with the 80C86, 80C88 and other microprocessors.

- The Intel 8259A Programmable Interrupt Controller handles up to eight vectored priority interrupts for the CPU. It is cascadable for up to 64 vectored priority interrupts without additional circuitry. It is packaged in a 28-pin DIP, uses NMOS technology and requires a single 5V supply. Circuitry is static, requiring no clock input. The 8259A is designed to minimize the software and real time overhead in handling multi-level priority interrupts. It has several modes, permitting optimization for a variety of system requirements. The 8259A is fully upward compatible with the Intel 8259. Software originally written for the 8259 will operate the 8259A in all 8259 equivalent modes.

3.6. Abbreviations:

PPI (8255)	: Programmable Peripheral Interface
PIT (8253/54):	: Programmable Timer.
PIC (8259)	: Programmable Interrupt Controller.
DMAC (8237/57)	: Direct memory access control.
USART (8251)	: Universal Serial Asynchronous Receiver & Transmitter
8279	: Keyboard & Display Interface
SRAM	: Static Random Access Memory
ROM	: Read only Memory
DRAM	: Dynamic Random Access Memory
EROM	: Electrical Read only Memory
E2ROM	: Electrical Erasable Read only Memory

3.7. Formulae: Nil

3.8. Key Definitions:

- 1) **Microprocessor:** A silicon chip that contains a CPU. In the world of personal computers, the terms microprocessor and CPU are used interchangeably.
- 2) **Memory:** Enables a computer to store, at least temporarily, data and programs.
- 3) **Central processing unit (CPU):** The heart of the computer, this is the component that actually executes instructions.
- 4) **Instruction set:** The set of instructions that the microprocessor can execute.
- 5) **Addressing modes:** The x86 instructions use five different operand types: registers, constants, and three memory addressing schemes. Each form is called an addressing mode. (i.e. way to access data/operand from memory location)

6)8085 Architecture:

It is 8 bit microprocessor i.e. it can read or write arithmetic operations on 8bit data at a time.

7)8085 Diagram: It is 40 pin IC and required +5v power supply.

8)BUS:A set of parallel connected lines is called as a bus.

9)System Bus: Microprocessor three system buses they are Address bus, data bus and control bus. The bus that connects the CPU to main memory on the motherboard. I/O buses, which connect the CPU with the systems other components, branch off of the system bus.

A communication pathway connecting two or more devices usually broadcast, often grouped. A number of channels in one bus e.g. 32 bit data bus is 32 separate single bit channels,Power lines may not be shown.

10)Data Bus:Carries data remember that there is no difference between “data” and “instruction” at this level Width is a key determinant of performance 8, 16, 32, 64 bit

11) Address bus: The bus over which the CPU sends out the address of the memory location is called as address bus.Identify the source or destination of data. e.g. CPU needs to read an instruction (data) from a given location in memory, Bus width determines maximum memory capacity of system e.g. 8080 has 16 bit address bus giving 64k address space.

12) Control Bus: The control bus consists of 4 to 10 parallel lines. The control bus is used for sending control signals to the memory & I/O devices.Control and timing information, Memory read/write signal, Interrupt request, Clock signals

13) Hardware, Software, and Firmware :When working around computers you hear the terms hardware, software and firmware. Hardware is the name given to the physical devices and circuitry of the computer. Software refers to the programs written for the computer. Firmware is the term given to the programs stored in ROMs or in other devices which keep their stored information when the power is turned off.

3.9. Course Content:

Lecture: 15

Memory Interfacing: SRAM, ROM

Learning Objective: In this lecture student will able to learn about Memory Interfacing: SRAM, ROM

3.9.1 Semiconductor Memory Fundamentals

In the design of all computers, semiconductor memories are used as primary storage for data and code.

They are connected directly to the CPU and they are the memory that the CPU asks for information (code or data)

- Among the most widely used are RAM and ROM

- Memory Capacity

- The number of bits that a semiconductor memory chip can store is called its chip capacity (bits or bytes)

- Memory Organization

- Each memory chip contains 2^x locations where x is the number of address pins on the chip
- Each location contains y bits, where y is the number of data pins on the chip
- The entire chip will contain $2^x * y$ bits
- Ex. Memory organization of $4K \times 4$: $2^{12} = 4096$ locations

Memory

- Each memory device has at least one chip select (CS) or chip enable (CE) or select (S) pin that enables the memory device.
- This enables read and/or write operations.
- Each memory device has at least one control pin.
- For ROMs, an output enable (OE) or gate (G) is present. The OE pin enables and disables a set of tristate buffers.
- For RAMs, a read-write (R/W) or

Memory Types

ROM (Read Only Memory)

ROM is the type of memory that does not lose its contents when power is turned off. It is also called Non-volatile memory.

PROM (Programmable Memory)

User programmable (one-time programmable) memory

If the information burned into PROM is wrong, it needs to be discarded since internal fuses are blown permanently.

Special equipment needed: ROM burner or ROM programmer

EPROM (Erasable Programmable ROM)

Allows making changes in the contents of PROM after it is burned

One can program the memory chip and erase it thousands of times

Erasing its contents can take up to 20 minutes; the entire chip is erased

All EPROM chips have a window that is used to shine ultraviolet (UV) radiation to erase its contents

Also referred to as UV-EPROM

EEPROM (Electrically Erasable ROM) 500,000 times

Method of erasure is electrical Moreover, one can select which byte to be erased

Cost per bit is much higher than for UV-EPROM

Flash Memory EPROM

First, the process of erasure of the entire contents takes less than a second, or one might say in a flash, hence its name: flash memory

When flash memory's contents are erased, the entire device is erased.

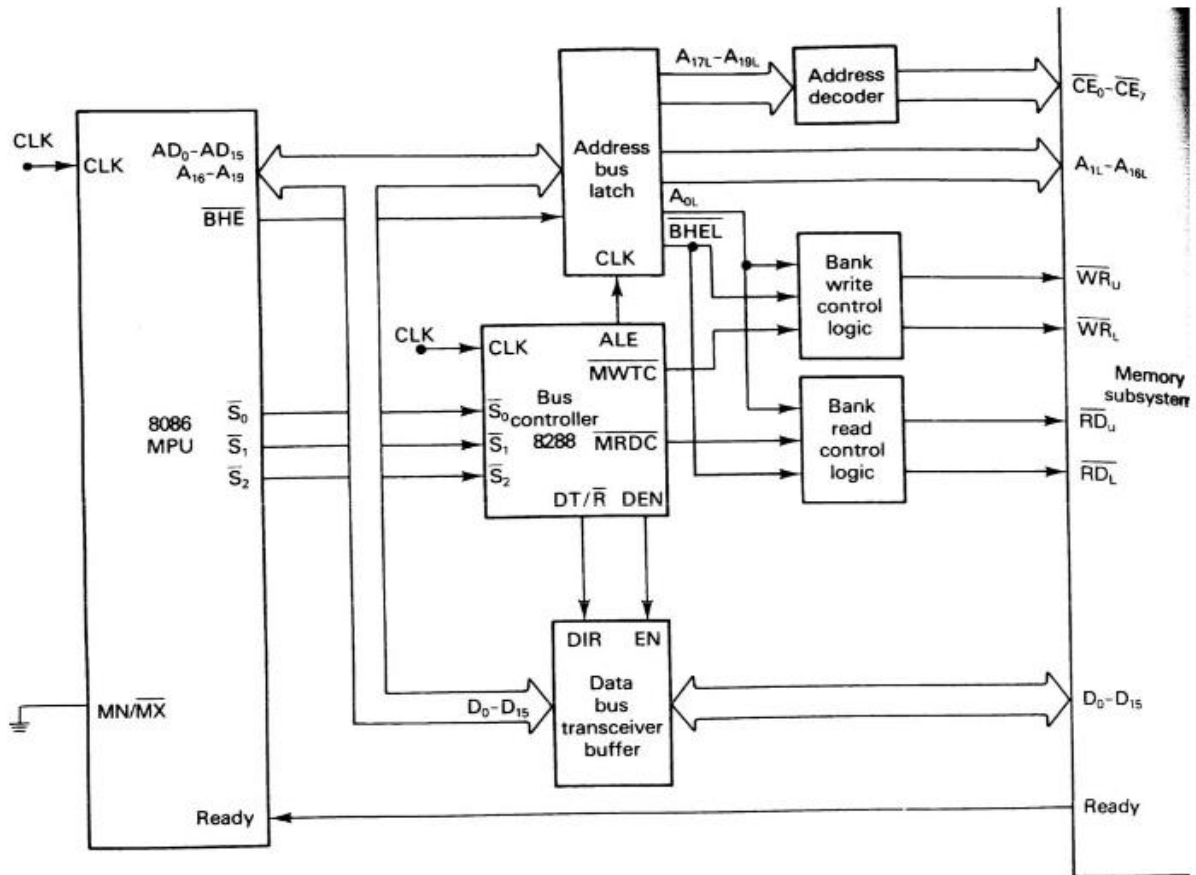
Even though flash memories are writeable, like EPROM they find their widest use in microcomputer systems for storage of firmware

RAM (Random Access Memory) infinite times

RAM memory is called volatile memory since cutting off the power to the IC will mean the loss of data.

Also referred to as R/W (Read and Write Memory)

Memory Interfacing



Let's check the take away from this lecture

1) Which of the following is a type of memory?

- i) ROM
- ii) RAM
- iii) EPROM
- iv) All of the above

L16. Exercise:

Q.1 Differentiate between Static and Dynamic RAM

Q.2 Differentiate between: RAM and ROM.

Questions/problems for practice:

Q. 3 List and explain the different types of memory

Learning from the lecture 'Memory Interfacing: SRAM, ROM:

Student will able to differentiate between SRAM and ROM.

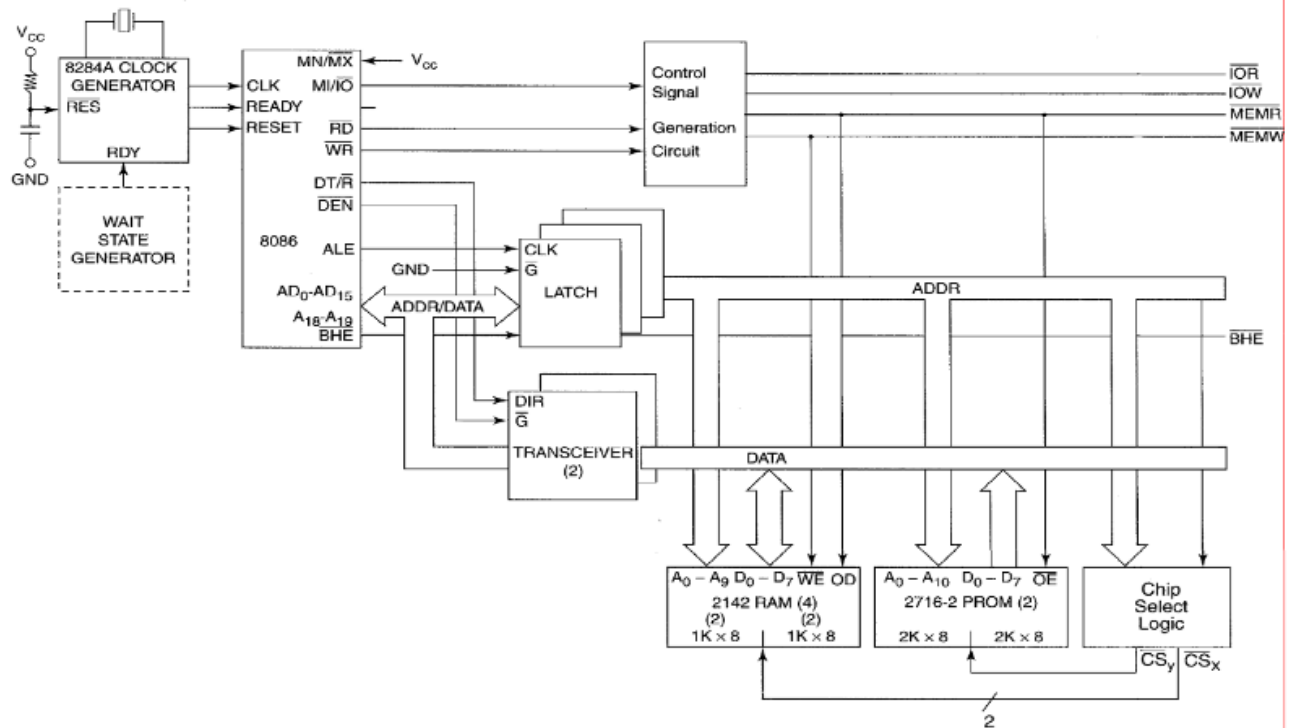
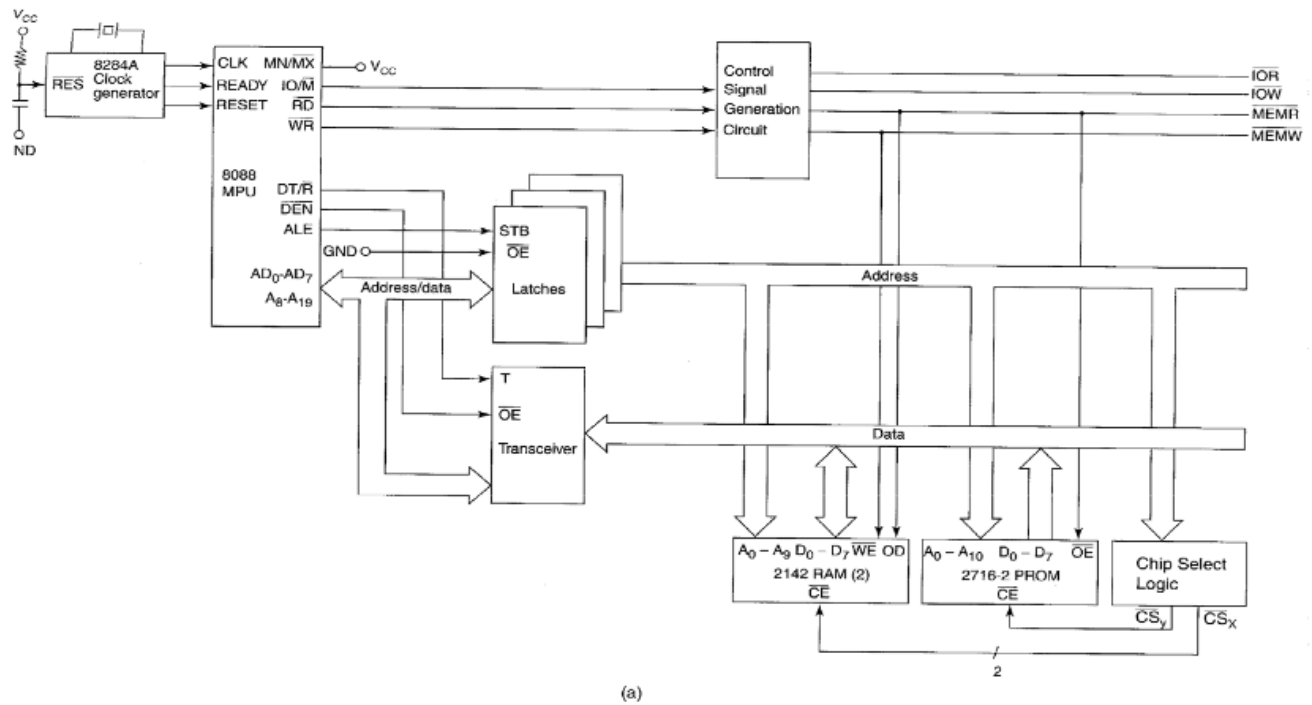
Lecture: 16

Memory Interfacing: SRAM, ROM: Minimum Mode

Learning objective: In this lecture students will able to understand Memory Interfacing: SRAM,

ROM in Minimum mode

3.9.4 Minimum Mode Configuration



Let's check the take away from this lecture

- 2) In 8086 microprocessor one of the following statements is not true.
- a) Coprocessor is interfaced in MAX mode
 - b) Coprocessor is interfaced in MIN mode
 - c) I/O can be interfaced in MAX / MIN mode
 - d) Supports pipelining

L17. Exercise

Q.4 Draw and explain Minimum mode configuration of 8086.

Questions/Problems for practice:

Q.5 Draw and explain Minimum mode configuration of 8088

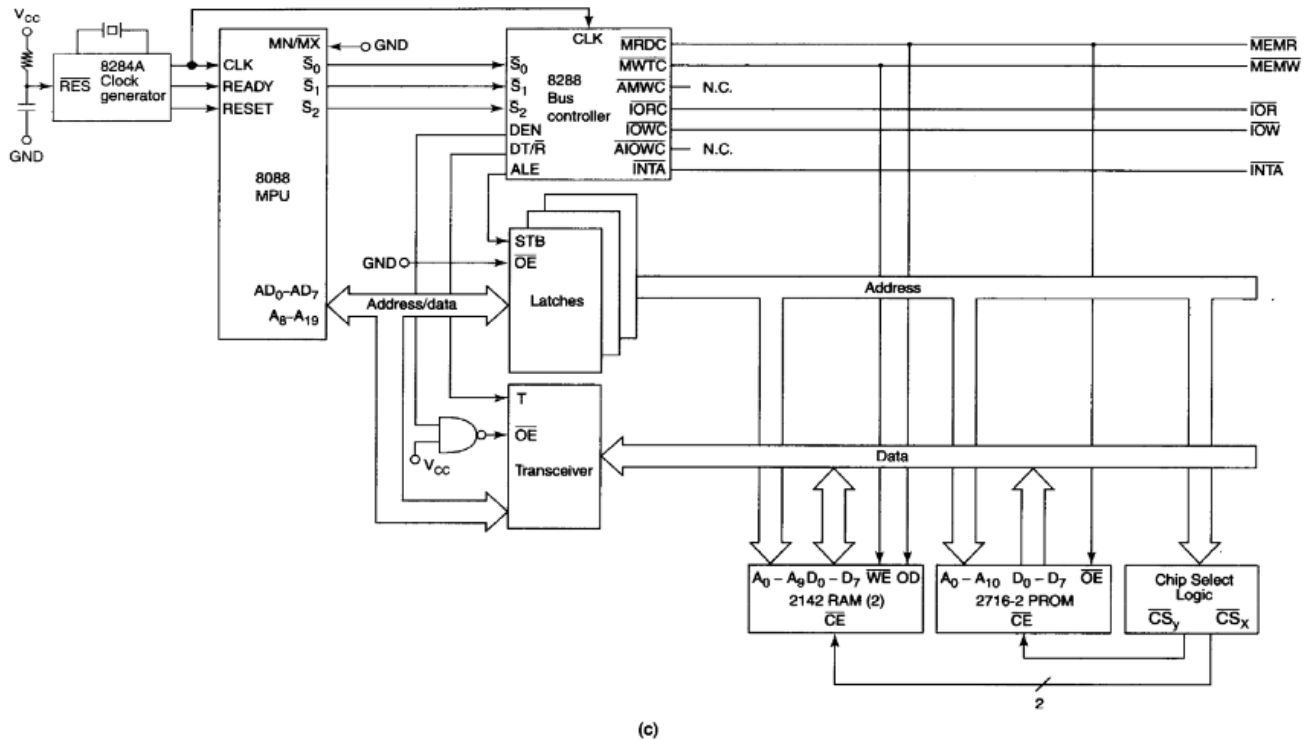
Learning from this lecture ‘*Memory Interfacing: SRAM, ROM: Minimum Mode*’:
Student will able to draw and explain Minimum mode configuration of 8086/8088.

Lecture: 17

Memory Interfacing: SRAM, ROM: Maximum Mode

Learning Objective: In this lecture student will able to learn about ***Memory Interfacing: SRAM, ROM: Maximum Mode***

1.9.5



Let's check the take away from this lecture

3. Which of the following statement is true for 8086 maximum mode?
 5. All the control signals are generated by 8086 itself
 6. All the control signals are generated by 8288 bus controller

L18 Exercise:

Q.6 Draw and explain maximum mode configuration of 8086

Questions/problems for practice:

Q.7 Draw and explain maximum mode configuration of 8088

Learning from the lecture 'Memory Interfacing: SRAM, ROM: Maximum Mode':

Student will able to draw and explain Maximum mode configuration of 8086/8088

Lecture: 18

Application, Control word format, Block Diagram of 8255 PPI and Interfacing with 8086/8088

Learning Objective: In this lecture students will able to understand Application, Control word format, Block Diagram of 8255 PPI and Interfacing with 8086/8088

1.9.7 Description:

8255 PPI is used to interface with keyboard devices. 8253 PIT is used to produce time and delay. 8259 PIC is used to generate interrupt whereas 8237 DMAC is used to transfer data between I/O

devices and memory. Also it contains interfacing of microprocessor with memory and microprocessor.

Features of 8255:

- 3 8-bit IO ports PA, PB, PC
- PA can be set for Modes 0, 1, 2. PB for 0,1 and PC for mode 0 and for BSR. Modes 1 and 2 are interrupt driven.
- PC has 2 4-bit parts: PC upper (PCU) and PC lower (PCL), each can be set independently for I or O. Each PC bit can be set/reset individually in BSR mode.
- PA and PCU are Group A (GA) and PB and PCL are Group B (GB)
- Address/data bus must be externally demux'd.
- TTL compatible.
- Improved dc driving capability

Background:

The Intel 8255 (or i8255) Programmable Peripheral Interface chip is a peripheral chip originally developed for the Intel 8085 microprocessor, and as such is a member of a large array of such chips, known as the MCS-85 Family. This chip was later also used with the Intel 8086 and its descendants. It was later made (cloned) by many other manufacturers.

This chip is used to give the CPU access to programmable parallel I/O, and is similar to other such chips like the Motorola 6520 PIA (Peripheral Interface Adapter) the MOS Technology 6522 (Versatile Interface adapter) and the MOS Technology CIA (Complex interface Adapter]] all developed for the 6502 family. Other such chips are the 2655 Programmable Peripheral Interface from the Signetics 2650 family of microprocessors, the 6820 PIO (Peripheral I/O) from the Motorola 6800 family, the western digital WDC 65C21, an enhanced 6520, and many others. The 8255 is perhaps most well-known for its use in the original IBM-PC's parallel printer port (now largely defunct and replaced by the USB standard, and considered a legacy port). However, most often the functionality the 8255 offered is now not implemented with the 8255 chip itself anymore, but is embedded in a larger VLSI chip as a sub function. The 8255 chip itself is still made, and is sometimes used together with a micro controller to expand its I/O capabilities.

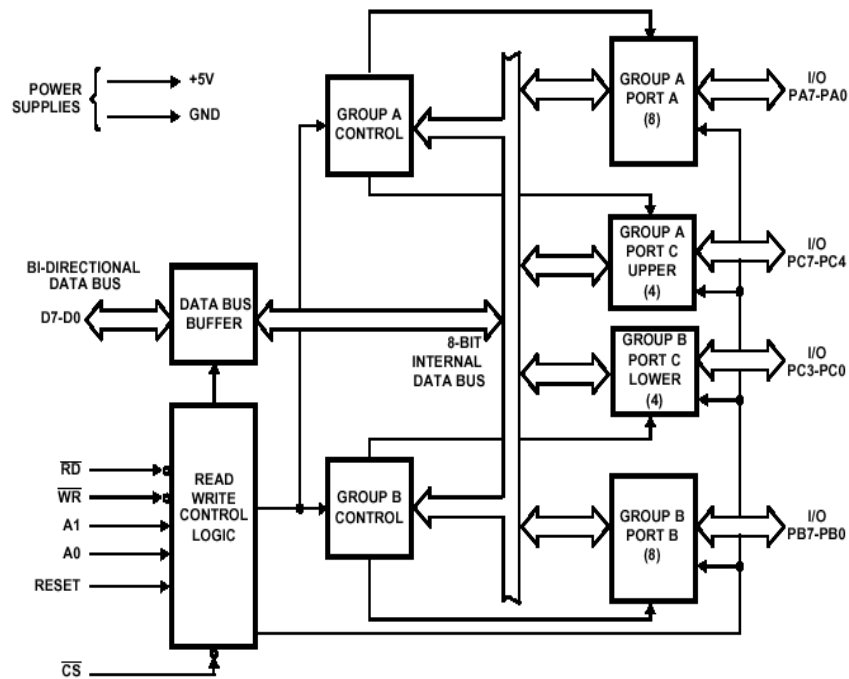


Figure 5.1: Block Diagram of the 8255 Programmable Peripheral Interface (PPI)

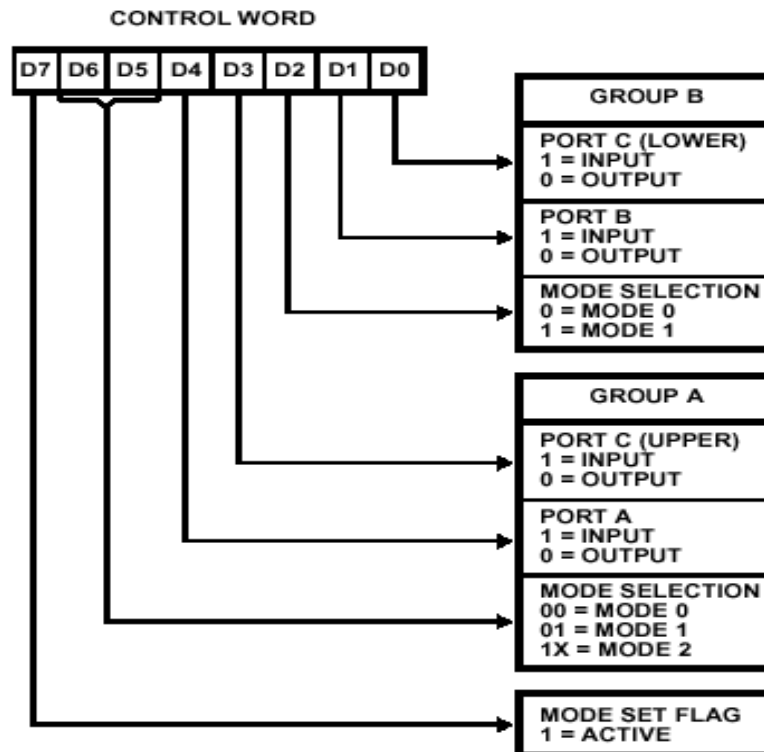


Figure 5.2 Mode

Group A and Group B Controls

The functional configuration of each port is programmed by the systems software. In essence, the CPU "outputs" a control word to the 8255. The control word contains information such as "mode", "bit set", "bit reset", etc., that initializes the functional configuration of the 8255. Each of the Control blocks (Group A and Group B) accepts "commands" from the Read/Write Control logic, receives "control words" from the internal data bus and issues the proper commands to its associated ports.

Ports A, B, and C

The 8255 contains three 8-bit ports (A, B, and C). All can be configured to a wide variety of functional characteristics by the system software but each has its own special features or "personality" to further enhance the power and flexibility of the 8255.

Port A One 8-bit data output latch/buffer and one 8-bit data input latch. Both "pull-up" and "pull-down" bus-hold devices are present on Port A.

Port B One 8-bit data input/output latch/buffer and one 8-bit data input buffer.

Port C One 8-bit data output latch/buffer and one 8-bit data input buffer (no latch for input). This port can be divided into two 4-bit ports under the mode control. Each 4-bit port contains a 4-bit latch and it can be used for the control signal output and status signal inputs in conjunction with ports A and B.

Let's check the take away from this lecture

5. The 8255 provide a total of pins that can be programmed as inputs or outputs.
- a. 24
 - b. 25
 - c. 20
 - d. 12

L20 Exercise:

Q.10 List and Explain features of 8255 PPI.

Questions/problems for Practice

Q.11 Explain 8255 & operation modes available in group A timing diagram and BSR mode

Learning from the lecture ‘Application, Control word format, Block Diagram of 8255 PPI and Interfacing with 8086/8088’:

Student will be able to understand the Application, Control word format, Block Diagram of 8255 PPI and Interfacing with 8086/8088

Lecture : 21

Interrupts

Learning Objective: In this lecture students will be able to learn state different types of Interrupts

Interrupt Structure of 8086:-

- An interrupt is a special condition that arises during the working of a μp .
- The μp services it by executing a subroutine called as the Interrupt Service Routine (ISR).
- There are 3 sources of interrupts for 8086:
 - i. External signal (Hardware Interrupts):** Interrupts that occur through the hardware pins NMI and INTR.
 - ii. Special instruction (Software Interrupts):** Occur when the μp encounters INT N instruction in the program.
 - iii. Condition produce by the program: (Exception):** 8086 is interrupted when some special condition occurs while executing certain instructions in the program.
Eg: Division by Zero automatically causes the divide error.

Memory Address	Table Entry	Vector Definition
3FE	CS 255	Vector 255
3FC	IP 255	
		User Available
82	CS 32	Vector 32
80	IP 32	
7E	CS 31	Vector 31
7C	IP 31	
		Reserved
16	CS 5	Vector 5
14	IP 5	
12	CS 4	Vector 4 - Overflow
10	IP 4	
0E	CS 3	Vector 3 - Breakpoint
0C	IP 3	
0A	CS 2	Vector 2 - NMI
08	IP 2	
06	CS 1	Vector 1 - Single- step
04	IP 1	
02	CS Value- Vector 0(CS 0)	Vector 0 - Divide Error
00	IP Value- Vector 0(IP 0)	
	2 Bytes	

Fig. 10 Interrupt Vector Table (IVT)

- The IVT contains ISR address for the 256 interrupts. Each ISR address is stored as CS and IP.
- As each ISR address is 4 byte (2 – CS and 2 – IP), each ISR address requires 4 locations to be stored. There are 256 interrupts: INT 0 ... INT 255. Therefore the total size of the IVT is $256 * 4 = 1\text{KB}$.
- The first 1KB of memory, address 00000H ... 003FFH, are reserved for the IVT.
- Whenever an interrupt INT N occurs, μP does $N * 4$ to get values of IP and CS for the ISR.

SOFTWARE INTERRUPTS

DEDICATED INTERRUPTS (INT 0 ... INT 4)

1. INT 0 (divide Error):

- This interrupt occurs whenever there is division error i.e. when the result of a division is too large to be stored.
- This condition normally occurs on division by zero.
- Its vector address is $4*0=00000\text{H}$.

2. INT 1 (Single Step):

- The μP execute this interrupt after every instruction if the TF is set in the flag register.
- It puts μP in single stepping Mode i.e. the μP pauses after executing every instruction.

- This is useful during debugging.
- Its ISR generally displays contents of all registers.
- Its vector address is $4 \times 1 = 00004H$.

3. INT 3 (Breakpoint Interrupt):

- The μP execute this interrupt internally in response to an interrupt on the NMI line.
- Its vector address is $4 \times 2 = 00008H$.

4. INT 4 (Overflow Interrupt):

- This interrupt is used to cause Breakpoints in the program.
- It occurs whenever the μP encounters the one-byte instruction INT.
- It is very useful in debugging large programs where single stepping is inefficient.
- The ISR for this interrupt generally displays the contents of all registers on the screen.
- Its vector address is $4 \times 3 = 0000CH$.

5. INT 4 (Overflow Interrupt):

- This interrupt occurs if the Overflow flag is set AND the μP executes the INTO instruction (Interrupt on overflow).
- It is used to detect overflow error in signed arithmetic operations.
- Its vector address is $4 \times 4 = 00010H$
- *Please note:* INT 0 INT 4 are called as dedicated interrupts as these are dedicated for the above-mentioned special conditions.

RESERVED INTERRUPTS

6. INT 5 ... INT 31:

- These levels are reserved by INTEL to be used in higher processors like 80386, Pentium etc.
- They are not available to the user.

USER DEFINED INTERRUPTS

7. INT 32 ... INT 255:

- These are user defined, software interrupts.
- ISRs for these interrupts are written by the users to service special conditions in the program.
- They are executed using the INT n instruction where the address is calculated as $n*4$.

HARDWARE INTERRUPTS

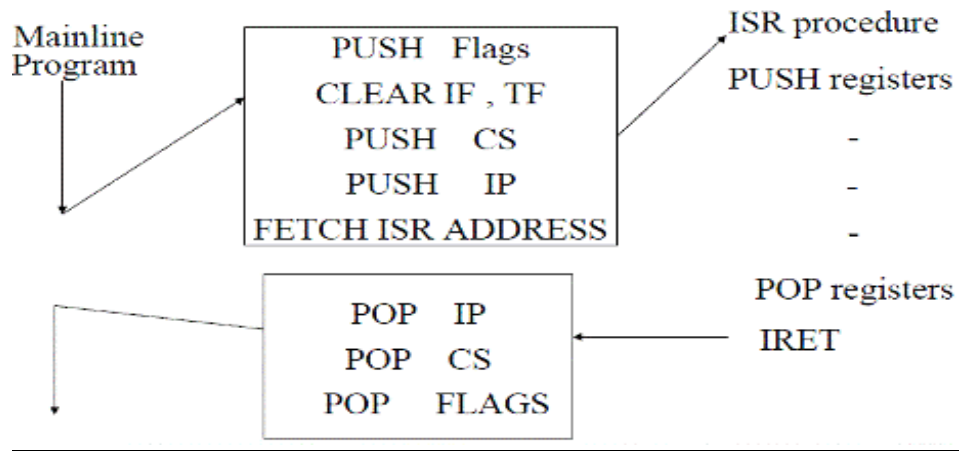
1. NMI (Non Maskable Interrupt):

- This is non-maskable, edge triggered, high priority interrupt.
- On receiving an interrupt on NMI line, the μP execute INT 2 and takes control to location $2*4 = 00008H$ in the Interrupt Vector Table.
- It reads 4 locations starting from this address to get the values for IP and CS, to execute the ISR.

2. INTR:

- This is a maskable, level triggered, low priority interrupt.
- On receiving an interrupt on INTR line, the μP executes 2 INTA(bar) pulses.
- First INTA(bar) pulse - - - the interrupting device calculates (prepares to send) the vector number.
- Second INTA(bar) pulse - - - the interrupting device sends the vector number "N" to the μP .
- Control shift to location pointed by Vector No * 4 in the IVT and IP and CS are loaded.
- It is masked if IF is cleared (by software through CLI instruction).
- It is unmasked if IF is set (by software through STI instruction).

RESPONSE TO ANY INTERRUPT - - - INT N



- i. The μP will PUSH Flag register into the Stack.
 $SS:[SP-1], SS:[SP-2] \leftarrow \text{Flag}$
 $SP \leftarrow SP-2$
- ii. Clear IF and TF in the flag register and thus disables INTR interrupt.
 $IF \leftarrow 0, TF \leftarrow 0$
- iii. PUSH CS into the stack.
 $SS:[SP-1], SS:[SP-2] \leftarrow CS$
 $SP \leftarrow SP-2$
- iv. PUSH IP into the Stack.
 $SS:[SP-1], SS:[SP-2] \leftarrow IP$
 $SP \leftarrow SP-2$
- v. Load new IP from the IVT
 $IP \leftarrow [N*4], [N*4+1]$
- vi. Load new CS from the IVT
 $CS \leftarrow [N*4+2], [N*4+3]$

Since CS and IP get new values, control shifts to the address of the ISR and the ISR thus begins. At the end of the ISR the μP encounters the IRET instruction and returns to the main program in the following steps:

RESPONSE TO IRET INSTRUCTION

- i. The μP will restore IP from the stack
 $IP \leftarrow SS:[SP], SS:[SP+1]$
 $SP \leftarrow SP+2$

- ii. The μP will restore CS from the stack.

$\text{Flag} \leftarrow \text{SS}:[\text{SP}], \text{SS}:[\text{SP}+1]$

$\text{SP} \leftarrow \text{SP}+2$

INTERRUPT PRIORITIES

<u>INTERRUPT</u>	<u>PRIORITY</u>	
	Simultaneous occurrence	To interrupt another ISR
Divide Error, INT n, INT 0	1(highest)	Can interrupt any ISR
NMI	2	
INTR	3	Cannot interrupt an ISR (IF, TF \leftarrow 0)
Single stepping	4(lowest)	

Priority in 8086 is of two types:

1.Simultaneous occurrence:

- When more than one interrupts occur simultaneously then, all s/w interrupts except single stepping, get highest priority.
- This is followed by NMI. Next is INTR. Finally, the lowest priority is of the single stepping interrupt.

eg: Assume the μP is executing a DIV instruction that causes a division error and simultaneously INTR occurs.

- Here INT 0 (division Error) will be serviced first i.e. its ISR will be executed, as it has higher priority, and then INTR will be serviced.

2. Ability to interrupt another ISR:

- Since software interrupt (INT N) are non-maskable, they can interrupt any ISR.
- NMI is also non-maskable hence it can also interrupt any ISR.
- But INTR and single stepping cannot interrupt another ISR as both are disabled before μP enters an ISR by $\text{IF} \leftarrow 0$ and $\text{TF} \leftarrow 0$.
- Eg: Assume the μP execute DIV instruction that causes a division error. So μP gets the INT 0 interrupt and now μP enters the ISR for INT 0. During the execution of this ISR, NMI and INTR occur.

- Here μP will branch out from the ISR of INT 0 and service NMI (as NMI is non-maskable).

After completing the ISR of NMI μP will return to the ISR for INT 0. INTR is still pending but the μP will not service INTR during the ISR of INT 0 (as $IF \leftarrow 0$). μP will first finish the INT 0 ISR and only then service INTR. Thus INTR and single stepping cannot interrupt an existing ISR.

Let's check the take away from this lecture

10. In 8086 the Interrupt Acknowledge is represented by _____ One MB is equal to:

- a) The amount of RAM in every computer
- b) 1 billion bytes
- c) 1024 KB
- d) 1 thousand bytes

11. In 8086 microprocessor the following has the highest priority among all type interrupts.

- a) NMI
- b) DIV 0
- c) TYPE 255
- d) OVER FLOW

L L9 Exercise:

Q.18 Draw and explain the Interrupt Vector of 8086 microprocessor.

Questions/problems for Practice

Q.19 Explain the interrupt structure of 8086.

Learning from the Lecture ' Interrupts:

Student will able to describe the interrupt structure of 8086 microprocessor

Lecture: 23

Application, Control word format, Block Diagram of 8259 PIC and Interfacing with 8086/8088

Learning Objective: In this lecture students will learn about Application, Control word format, Block Diagram of 8259 PIC and Interfacing with 8086/8088

1.9.7

Features of 8259:

- 8 levels of interrupts.
- Can be cascaded in master-slave configuration to handle 64 levels of interrupts.
- Internal priority resolver.
- Fixed priority mode and rotating priority mode.
- Individually maskable interrupts.
- Modes and masks can be changed dynamically.
- Accepts IRQ, determines priority, checks whether incoming priority > current level being serviced, issues interrupt signal.
- In 8085 mode, provides 3 byte CALL instruction. In 8086 mode, provides 8 bit vector number.
- Polled and vectored mode.
- Starting address of ISR or vector number is programmable.
- No clock required.

Command Words of 8259A:

The command words of 8259A are classified in two groups, viz. initialization command words (ICWs) and operation command words (OCWs). Initialization Command Words (ICWs) Before it starts functioning, the 8259A must be initialized by writing two to four command words into the respective command word registers. These are called as initialization command words (ICWs). If $A0 = 0$ and $D4 = 1$, the control word is recognized as ICW1. It contains the control bits for edge/level triggered mode, single/cascade mode, call address interval and whether ICW4 is required or not, etc. If $A0 = 1$, the control word is recognized as ICW2. The ICW2 stores details regarding interrupt vector addresses. The initialisation sequence of 8259A is described in form of a flow chart. The bit functions of the ICW1 and ICW2 are self-explanatory as shown in Fig. Once ICW1 is loaded, the following initialization procedure is carried out internally.

(a) The edge sense circuit is reset, i.e. by default 8259A interrupts are edge sensitive.

(b) IMR is cleared.

(c) IR7 input is assigned the lowest priority.

(d) Slave mode address is set to 7.

(e) Special mask mode is cleared and status read is set to IRR.

(f) If $IC4 = 0$, all the functions of ICW4 are set to zero. Master/slave bit in ICW4 is used in the buffered mode only.

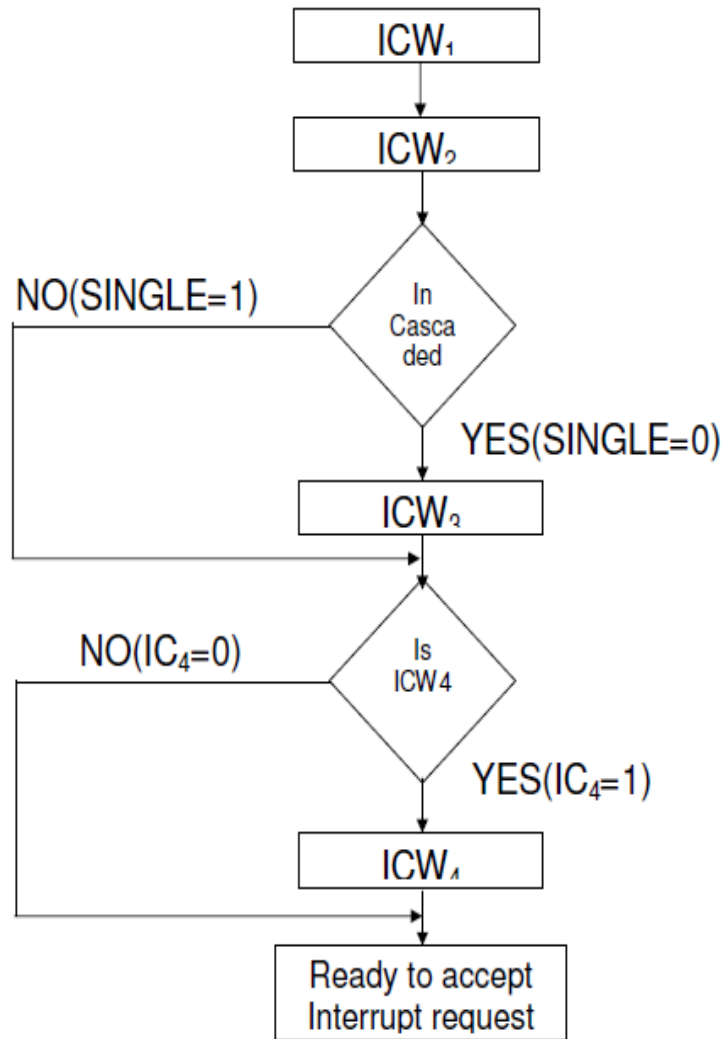


Fig1.3. Initialisation Sequence of 8259A

In an 8085 based system, A15 – A8 of the interrupt vector address are the respective bits of ICW2. In 8086/88 based system A15 – A11 of the interrupt vector address are inserted in place of T7 - T3 respectively and the remaining three bits (A8, A9 and A10) are selected depending upon the interrupt level, i.e. from 000 to 111 for IR0 to IR7.

ICW1 and ICW2 are compulsory command words in initialization sequence of 8259A as is evident from Fig. 1.3, while ICW3 and ICW4 are optional. The ICW3 is read only when there are more than one 8259As in the system, i.e. cascading is used (SNGL = 0). The SNGL bit in ICW1 indicates whether the 8259A is in cascade mode or not. The ICW3 loads an 8-bit slave register. Its detailed functions are as follows.

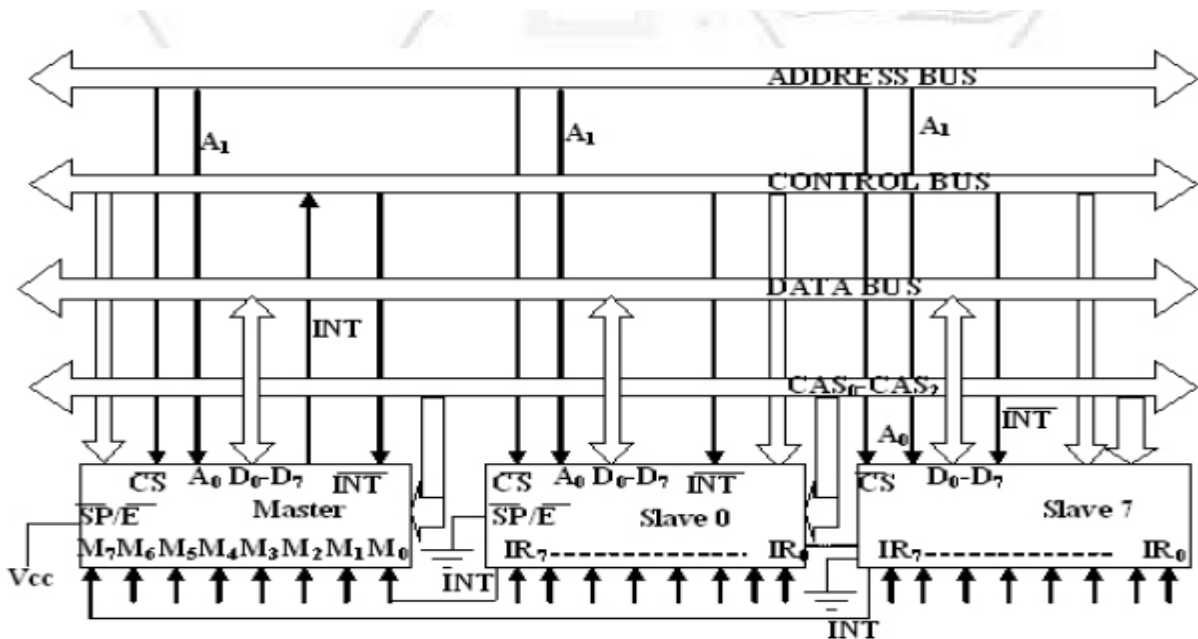
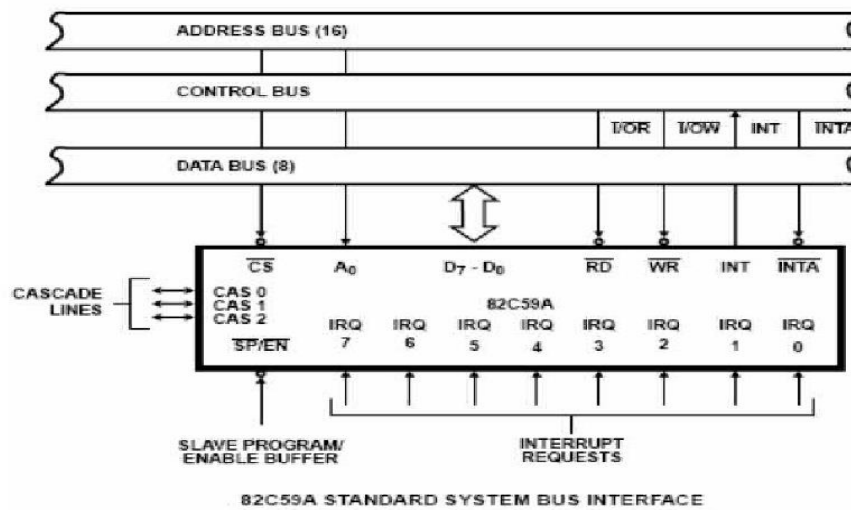
In master mode [i.e. $SP = 1$ or in buffer mode $M/S = 1$ in ICW4], the 8-bit slave register will be set bit-wise to '1' for each slave in the system, as shown in Fig. 1.5. The requesting slave will then release the second byte of a CALL sequence.

In slave mode [i.e. $SP=0$ or if $BUF=1$ and $M/S=0$ in ICW4] bits D2 to D0 identify the slave, i.e 000 to 111 for slave 1 to slave 8. The slave compares the cascade inputs with these bits and if they are equal, the second byte of the CALL sequence is released by it on the data bus.

8259A PIC- BLOCK DIAGRAM

It includes 8 blocks.

- Control logic
- Read/Write logic
- Data bus buffer
- Three registers (IRR,ISR and IMR)
- Priority resolver
- Cascade Buffer



Cascade mode : Interfacing of 3-PIC 8259 with 8086.

10. Using one 8259 the single INTR input of the 8086 processors can be expanded to accommodate up to prioritized interrupt inputs.

- a. 8
- b. 2
- c. 4
- d. 1.

L23 Exercise:

Q.16 What are the features of 8259?

Question/ problems for practice

Q.17 Draw block diagram of 8259 PIC

Q.18 Draw diagram to interface 8259 PIC with system bus

Learning from the Lecture ‘ Application, Control word format, Block Diagram of 8259 PIC and Interfacing with 8086/8088:

Student will able to explain Application, Control word format, Block Diagram of 8259 PIC and Interfacing with 8086/8088

Lecture: 24

Introduction to 8087 Math Coprocessor

Learning Objective: In this lecture students will learn about Introduction to 8087 Math Coprocessor

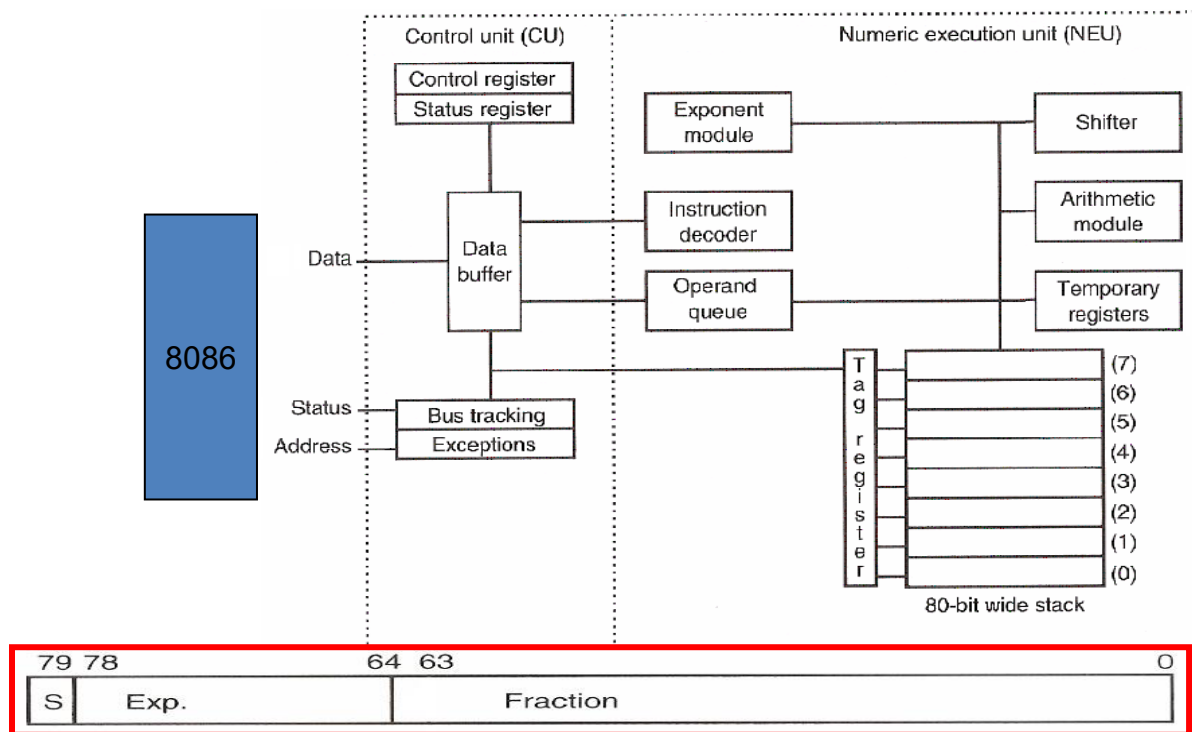
1.9.7 The Math Coprocessor:

(Numeric Data Processor (NDP))

- The 8086 performs integer math operations
- Floating point operations are needed, e.g. for Sqrt (X), sin (x), etc.
- These are complex math operations that require large registers, complex circuits, and large areas on the chip
- A general data processor avoids this much burden and delegates such operations to a processor designed specifically for this purpose - e.g. math coprocessor (8087) for the 8086
- The 8086 and the 8087 coprocessors operate in parallel and share the busses and memory resources
- The 8086 marks floating point operations as ESC instructions, will ignore them and 8087 will pick them up and execute them

The 8087 Coprocessor: Organization

- CU and NEU units
- Eight 80-bit FP Registers
- Supports 68 FP (ESC) instructions
- Speeds up 8086 performance on FP operations by a factor of 50-100 time
- 8087 Tracks activities of the 8086 by monitoring:
 - Bus status (S0-S2 bits)
 - Queue status (QS0,1)
 - Instruction being fetched (to check if its an ESC instruction)
- Synchronize with WAIT using the BUSY-#TEST signals



8086 Maximum mode outputs for NDP Connection

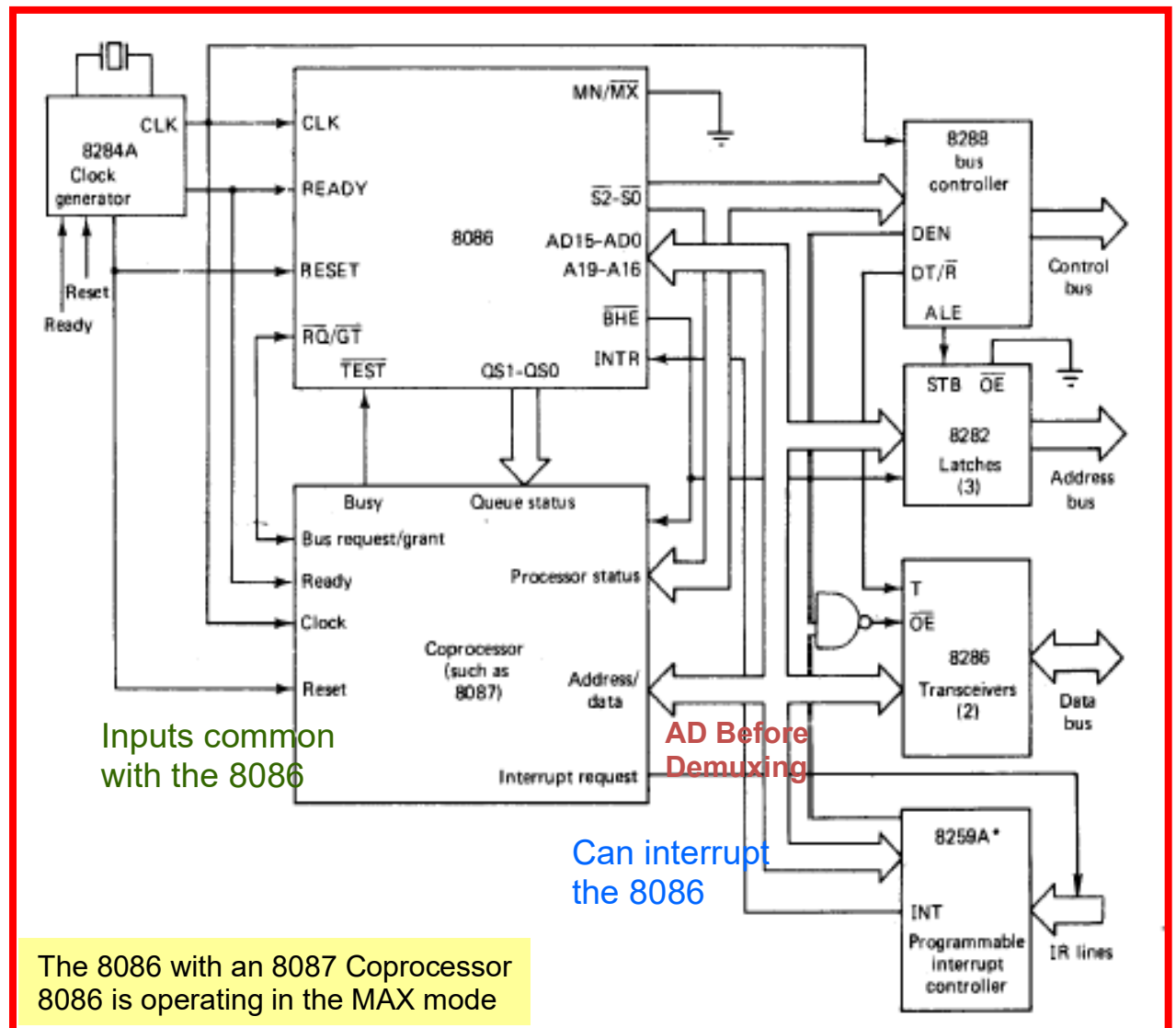
- **Bus Status Outputs S0-S2:**

Status bits that encode the type of the current bus cycle

- **Bus Request/Grant Outputs RQ0/GT0:**

Allow 8087 to request use of the bus,
e.g. for DMA memory access

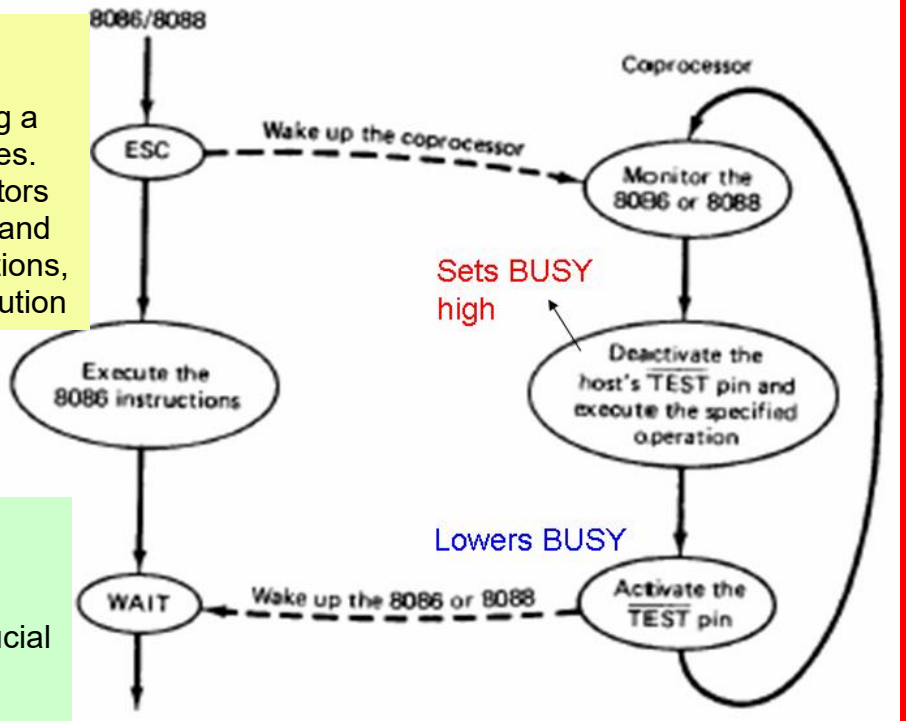
- **Queue Status Outputs QS1,QS0:**



- Synchronization between 8086 & the 8087 Coprocessor

The assembler marks all FP instructions as ESC instructions having a special range of opcodes. The Coprocessor monitors the 8086 bus activities and intercepts such instructions, captures them for execution

WAIT instructions can be used to halt the 8086 to ensure that the 8087 has finished a crucial step, e.g. storing a result in memory.



- For use by coprocessors that receive their instructions via ESC prefix.
- Allow the coprocessor to track the progress of an instruction through the 8086 queue and help it determine when to access the bus for the escape op-code and operand.
- Indicate the status of the internal instruction queue as given in the table:

QS1	QS0	
0	0	Queue is idle
0	1	First byte of opcode from queue
1	0	Queue is empty
1	1	Subsequent byte of opcode from queue

L24 Exercise:

Q.19 Explain Interfacing of 8087 NDP with 8086.

Question/ problems for practice

Q.20 Draw block diagram of 8087NDP

Learning from the Lecture ‘ Introduction to 8087 Math Coprocessor :

Student will able to explain Introduction to 8087 Math Coprocessor and its interfacing

1.10 Learning Outcomes:

1. Know:

- d) Describe the working of 8259PIC. How to initialize and make 8259 Programmable to accept software and hardware interrupt requests in different operating modes .Also Illustrate the working of cascade mode.
- e) Explain the functionality of coprocessor and how it communicate with 8086 processor for execution of floating point operations.

2. Comprehend:

- a) To Analyze the working of 8255 PPI. How it is used for 8 or 16 bit data transfer to /from peripheral devices and processor
- b) Distinguish between minimum and maximum mode and Understand the concept of ODD and EVEN memory bank and how to interface SRAM and DRAM with 8086/8088 in minimum and maximum mode.

3. Apply, analyze and synthesize:

- a) Illustrate the working of 8253 and show demonstration for different applications like Square wave generator, Interrupt on terminal, software trigger etc.
- b) Design microprocessor based system for given specification using minimum mode and maximum mode

1.11. Multiple Choice Questions

- 1. If the size of the segment is 64 kb, what will be the starting and ending off set addresses of it
 - a) 0000H to 7FFFH
 - b) 0000H to FFFFH
 - c) 8000H to FFFFH
 - d) 00000H to FFFFFH

- 2. A 20-bit address bus can locate .

- A. 1,048,576 locations B. 2,097,152 locations
C. 4,194,304 locations D. 8,388,608 locations
3. The number of address and data lines of 8086_____.
- a) 8 and 8
 - b) 16 and 16
 - c) 20 and 16
 - d) 16 and 20
4. ____ bit in ICW1 indicates whether the 8259A is cascade mode or not? a) LTIM=0 b) LTIM=1 c) SNGL=0 d) SNGL=1
5. 8086 is interfaced to two 8259s (Programmable interrupt controllers). If 8259s are in master slave configuration the number of interrupts available to the 8086 microprocessor is
- a) 8 b) 16 c) 15 d) 64
6. Access time is faster for
- a) ROM b) SRAM c) DRAM
7. ROM is generally known to be ?
- a). Non-volatile memory b). Volatile memory c). Both d). None of the above.
8. 1). Which type of RAM is better for high performance systems?
- a) Static b). Dynamic c). Both d). None.
9. Programmable peripheral input-output port is another name for
- a) serial input-output port
 - b) parallel input-output port
 - c) serial input port
 - d) parallel output port
10. Port C of 8255 can function independently as
- a) input port
 - b) output port
 - c) either input or output ports
 - d) both input and output ports
11. All the functions of the ports of 8255 are achieved by programming the bits of an internal register called
- a) data bus control
 - b) read logic control
 - c) control word register
 - d) none of the mentioned

12. The port that is used for the generation of handshake lines in mode 1 or mode 2 is
 - a) port A
 - b) port B
 - c) port C Lower
 - d) port C Upper
13. The disadvantage with SRAM is:
 - a). size b). power consumption c). periodic refresh is needed d). there is no disadvantage with SRAM.
14. Which type of RAM is faster?
 - a). STATIC b). Dynamic c). It depends on the situation of READ or WRITE d). None of the above.
15. The pin that clears the control word register of 8255 when enabled is
 - a) CLEAR
 - b) SET
 - c) RESET
 - d) CLK

1.12 Short Answer questions

Q.1 Draw the flowchart of Initialization of 8259 PIC.

Q.2 List the different operating modes of 8259PIC.

Q.3 Draw block diagram of 8259PIC.

Q.4 Draw the command word format of 8255PPI.

Q.5 List the applications of different Peripheral Devices.

1.12. Long Answer question

Q.1. Draw & explain the block diagram of 8255A.

Q.2 Draw and explain interface of 8255 with 8086.

Q.3 Design a 8086 based system consisting of the following (Minimum mode) :-
microprocessor working at 8 MHz

8086

1. EPROM of 32 KB using 16 KB device

2. SRAM of 64 KB using 32 KB devices

1 input, 1 output port (both 16 bits), interrupt devices. Explain the design.

Q.4 Design a 8086 based microprocessor system with the following specifications:-

(A) 8086 is working at 8 MHz.

(B) 64KB EPROM using 32 KB devices.

(C) 16KB SRAM using 8 KB devices.

(D) 8087 – coprocessor connected

Q.5 Interface 8259 PIC with 8086 in cascade mode. (No. of 8259=3 in the system).

1.15. References

6. Microprocessors and Interfacing ,Douglas V Hall,T ata Mc Graw Hill
7. http://www.telnet7.net/articles/not_mine/8086_achitecture.htm
8. The 8086/8088 Family, John Uffenbuck, Pearson Media, LPE

Self-Assessment

Self-Evaluation

Name of Student		
Class		
Roll No.		
Subject		
Module No.		
S.No		Tick Your choice
11.	Do you understand the architecture of 8086	<input type="radio"/> Yes <input type="radio"/> No
12.	Do you understand the functions of pins	<input type="radio"/> Yes <input type="radio"/> No
13.	Do you understand the operating modes of 8086	<input type="radio"/> Yes <input type="radio"/> No
14.	Do you able to expalin memory segmentation?	<input type="radio"/> Yes <input type="radio"/> No
15.	Do you understand module ?	<input type="radio"/> Yes, Completely. <input type="radio"/> Partialy. <input type="radio"/> No, Not at all.

Module: 4

INTEL 80386DX PROCESSOR

4.1. Motivation:

At the completion of the subject, students should be able to:

- Describe the overview architecture of advanced microprocessor systems. identify and analyze Real-Mode software
- Identify and analyze Protected-Mode software architecture of the 80386DX microprocessors.
- To design microprocessor based system.
- Detail study of software & hardware architecture of Intel 80386DX processor.
- Features of Intel 80386Dx processor.

4.2. Syllabus:

Lecture	Content	Duration	Self Study
23	Intel 80386dx Processor: Detailed study of Block diagram	2 Lecture	2 hours
24	Programming model / Register Organization	2 Lecture	2 hours
25	Operating Modes: Real Mode, Protected Mode And Virtual 8086 Mode	1 Lecture	2 hours
26	Address translation mechanism in protected mode	1 Lecture	2 hours
27	Segmentation	1 Lecture	2 hours
28	Paging	1 Lecture	2 hours

4.3. Weightage: 20 Marks

4.4. Learning Objectives: Students should be able to-

1. List the advanced features of Intel 80386Dx processor such as ALU size, segmentation and paging , operating modes and protection mechanism etc. and compare the performance of 80386Dx with its predecessors (R)
2. Differentiate between Intel 80386Dx and 80386Sx w.r.t. address bus, data bus, cache

memory , physical memory etc. (AN)

3. Describe software architecture (Programmer's model / Register set) and Hardware architecture (Block Diagram) of Intel 80386Dx processor (U)

4. Recall Logical, Linear and physical addresses and advantages and drawbacks of memory segmentation and paging mechanism.(R)

5. Compare different operating modes of Intel 80386Dx processor w.r.t. size of Physical Memory, segmentation and paging mechanism, enter and exit from operating modes etc.(E)

6. Give example what is need of protection mechanism in multitasking environment during segment and page level translation mechanism. (U)

4.5. Theoretical Background:

The objective of this course is: To study microprocessor basics & the fundamental principles of architecture related to advanced microprocessors & develop understanding of the architectures of advanced microprocessors and to acquire the skills in the programming and applications of these processors.

- To design microprocessor-based systems and write program in assembly language.
- Operating modes (Real mode, Protected mode, Virtual 8086 mode) of Intel 80386Dx processor
- Special features of protected mode

4.6. Abbreviations:

- PF: PreFetch
- D1: Instruction Decode 1
- D2: Instruction Decode 2
- EX: Execution Stage
- ER: Error Reporting
- WB: Write Back
- FP: Floating Point
- MESI: Modified Exclusive Shared Invalid
- INVD: Invalidate
- WBINVD: Write Back and Invalidate

- GDT : Global Descriptor Table
- LDT : Local descriptor Table
- IDT :Interrupt Descriptor Table
- IVT : Interrupt Vector Table
- GDTR : Global Descriptor Table Register
- LDTR : Local Descriptor Table Register
- IDTR : Interrupt Descriptor Table Register
- PE : Protection Enable
- TLB : Translation Look aside Buffer
- PTE :Page Table Entry
- PDE :Page Directory Entry
- PL : Privilege level
- CR : Control Register
- MMU : Memory Management Unit
- DR : Debug Register
- SMM : System Management Mode
- 80386DX :80386 Double Execution
- 80386SX :80386 Single Execution
- TI : table Indicator
- PDBR : Page Table Base Register

4.7. Formulae:

4.8. Key Definitions:

Microprocessor: A microprocessor is a processor (or Central Processing Unit, CPU) fabricated on a single integrated circuit.

Task: Group of instructions written to perform a specific operation is called as a task.

Segment: Block of consecutive memory locations having same properties or attributes as called as a segment.

Versions of Intel 80386 processor:

- 80386DX - this CPU could work with 16-bit and 32-bit external buses.
- 80386SX - low cost version of the 80386. This processor had 16 bit external data bus and 24-bit external address bus.
- 80386SL - low-power microprocessor with power management features, with 16-bit external data bus and 24-bit external address bus. The processor included ISA bus controller, memory controller and cache controller.

4.9. Course Content:

Lecture: 23 & 24

Detailed study of Features and Block diagram

Learning Objective: In this lecture student will able to learn about architecture of Intel 80386DX.

4.9.1 Introduction–

Features of 80386DX: Introduced in 1985 by Intel.

- 1) True 32 bit microprocessor .
- 2) Size of ALU 32bit
- 3) Size of data bus 32bit
- 4) Size address bus 32bit
- 5) Size of physical memory 4GB (2^{32})
- 6) Size of virtual memory-64 TB (2^{46}) (Addition of virtual memory leads to performance degradation hence to increase the performance , Level 2 SRAM cache was added)
- 7) Operating modes: -
 - a) Real mode b) Protected mode c)Virtual 8086 mode
- 8) Support protection mechanism
- 9) Support paging mechanism
- 10) Support multitasking
- 11) Compatible with all it's predecessors.
- 12) Memory management unit: The MMU provides Virtual memory, paging and four levels of protection. The concept of paging enables it to organize available physical memory in terms of pages of 4K under segmented memory.
- 13) It is a 32 bit processor that supports 8/16/32 bit data operands.
- 14) 32 bit internal registers. New category of registers Control, Test and Debug

15) Instruction set upward compatible to its predecessors. New instructions deal with protection mechanism, memory segmentation and paging MMU.

16) Clock frequency starting from 16 MHz Different versions have 33 MHz, 66 MHz. Available as 80386 DX, 80386SX, 80386 SL etc.

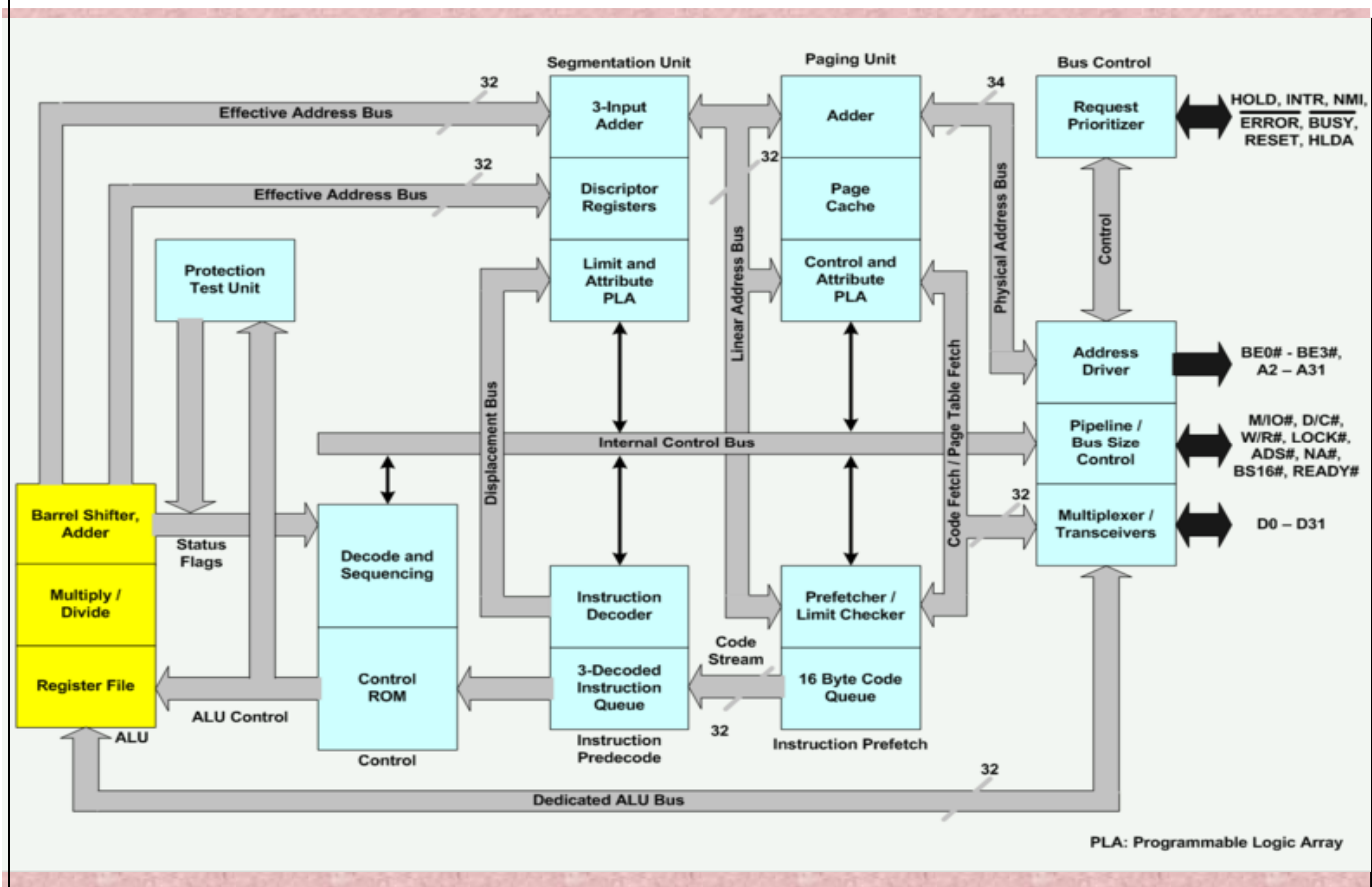
17) Data types supported are byte, word, double word, unpacked BCD, Packed BCD, Long integer(64), Short real(32), long real (64), extended real (80) (Real are for floating point).

18) Available in 132 pin grid array package, some are in 100 pin etc.

19) It has 11 addressing modes.

4.9.2 Software & Hardware architecture of 80386DX processor:

a) Detailed Block diagram of 80386DX processor:-



Description :

•The Internal Architecture of 80386 is divided into 3 sections.

1. Central processing unit
2. Memory management unit
3. Bus interface unit

1) Central processing unit :- is further divided into Execution unit and Instruction unit.

•**Execution unit** has 8 General purpose and 8 Special purpose registers which are either used for handling data or calculating offset addresses.

• **Instruction unit :**

- Decodes the opcode bytes received from the 16-byte instruction code queue and arranges them in a 3- instruction decoded instruction queue.
- After decoding them pass it to the control section for deriving the necessary control signals.
- The barrel shifter increases the speed of all shift and rotate operations.
- The multiply / divide logic implements the bit-shift-rotate algorithms to complete the operations in minimum time.
- Even 32- bit multiplications can be executed within one microsecond by the multiply / divide logic.

2) The Memory management unit :- consists of a Segmentation unit and a Paging unit.

- **Segmentation unit:**

- allows the use of two address components, viz. segment and offset for revocability and sharing of code and data.
- Segmentation unit allows segments of size 4Gbytes at max.

- **Paging unit :**

- organizes the physical memory in terms of pages of 4kbytes size each.
- Paging unit works under the control of the segmentation unit, i.e. each segment is further divided into pages. The virtual memory is also organizes in terms of segments and pages by the memory management unit.
- The Segmentation unit provides a 4 level protection mechanism for protecting and isolating the system code and data from those of the application program.
- Paging unit converts linear addresses into physical addresses.

3) The Bus control unit :-

- Has a prioritizer to resolve the priority of the various bus requests.
- This controls the access of the bus. The address driver drives the bus enable and address signal A0 – A31. The pipeline and dynamic bus sizing unit handle the related control signals.
- The data buffers interface the internal data bus with the system bus.

Let's check the take away from this lecture

- 1) What is size of address bus in 80386DX processor?
1)32bit_ 2)16bit 3) 4)64bit 4)24bit
- 2) Segmentation unit allows segments of _____size at maximum.
a) 4Gbytes b) 6Mbytes c) 4Mbytes d) 6Gbytes

L21. Exercise:

Q.1 Define segmentation and paging.

Q.2 List the features of Intel 80386DX processor.

Questions/problems for practice:

Q. 3 Differentiate between logical address , linear address and physical address.

Learning from the lecture ‘Detailed study of Block diagram, Signal interface, Bus cycles’:

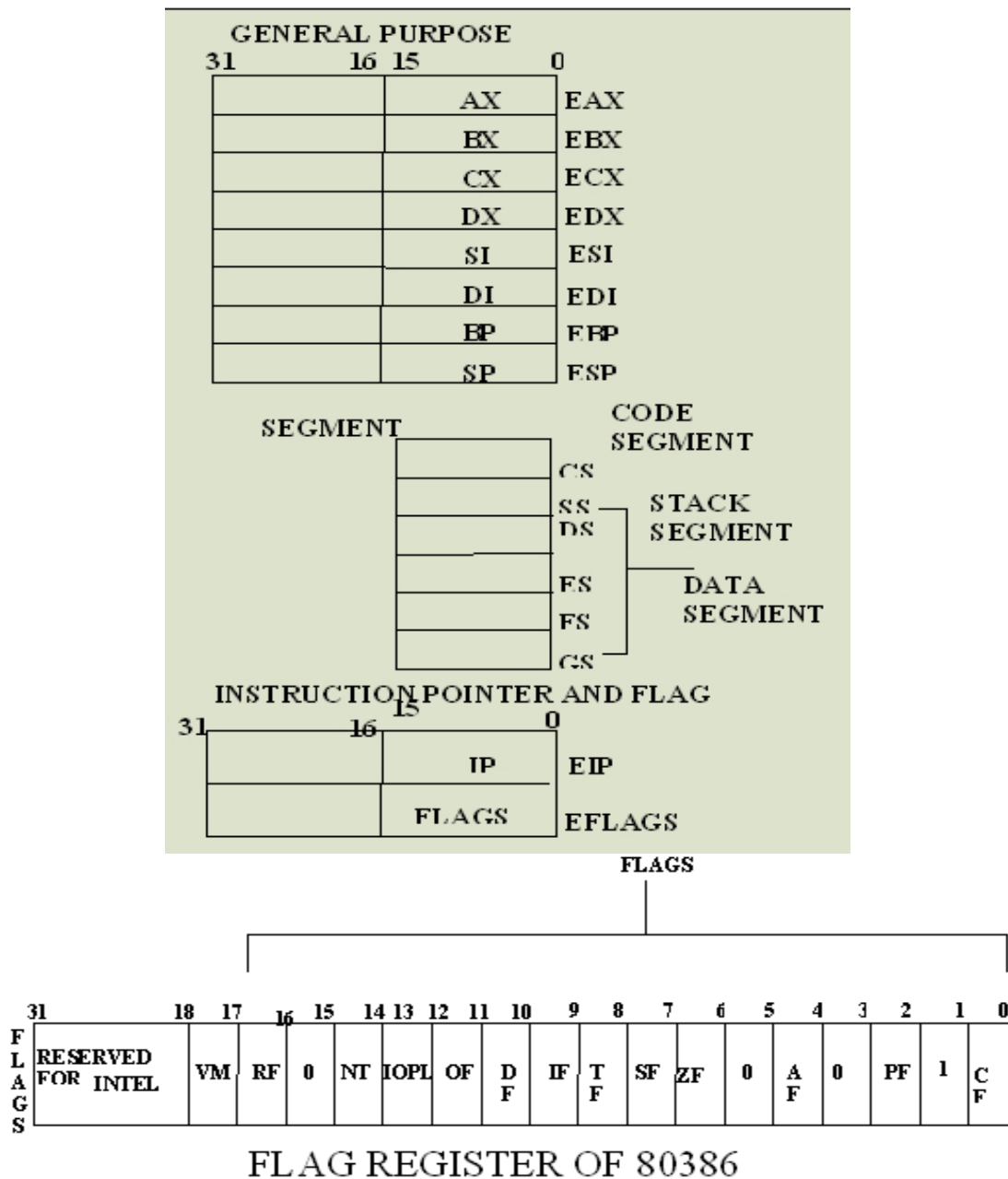
Student will able to list and explain the advanced features of Intel 80386DX and describe the working of Intel 80386Dx processor.

Lecture: 25 & 26

Register Set/ Programmer's model/ S/W architecture of Intel 80386DX processor

Learning objective: In this lecture students will able to software architecture of Intel 80386DX.

4.9.3 Register set:



-Flag register is a group of flip flops.

-In Intel 80386 13 flag bits are active among these

-First 9 flag bits same as 8086 and 4 new flag bits are added in 80386DX.

CF-Carry Flag

PF-Parity Flag

AF-Auxiliary Flag

ZF- Zero Flag

SF- Sign Flag

TF-Trap Flag

IF-Interrupt Flag

DF- Direction Flag

OF-Overflow Flag

-New flag bits:

IOPL: I/O –O/P privilege level

It gives minimum PL for the execution of IOPL sensitive instructions such as IN & OUT.

NT:Nested Task

If NT=1 it means valid backlink is existing in TSS (Task State Segment)

Nested task means that, in multitasking after switching from task A to task B we can return to task A instead of going to task C (i.e. task is nested)

RF: (Resume Flag)

If RF=1; debug exception is ignored

RF=0; debug exception is generated

This flag is used with the debug register breakpoints. It is checked at the starting of every instruction cycle and if it is set, any debug fault is ignored during the instruction cycle.

VM: Virtual 8086 mode

VM=1 ; virtual 8086 mode

VM=0 ;Protected mode

If this flag is set, the 80386 enters the virtual 8086 mode within the protection mode. This is to be set only when the 80386 is in protected mode. In this mode, if any privileged instruction is executed an exception 13 is generated. This bit can be set using IRET instruction or any task switch operation only in the protected mode.

System/Memory Management Registers (Associated with Protection):

- 1) GDTR : **G**lobal **D**escriptor **T**able **R**egister.48 bit register.
- 2)IDTR : **I**nterrupt **D**escriptor **T**able **R**egister.48 bit register.
- 3) LDTR: **L**ocal **D**escriptor **T**able **R**egister.16 bit register.
- 4)TR: **T**ask **R**egister . 16 bit register.

Function:

GDTR and LDTR points to the segment descriptor tables, GDT and LDT.

IDTR points to the table of entry points for interrupt handlers.

TR points to the information needed by the processor to define the current task.

GDTR:

Base (32bit)	Limit (16bit)
--------------	---------------

IDTR:

Base (32bit)	Limit (16bit)
--------------	---------------

LDTR:

Selector (16bit)

TR:

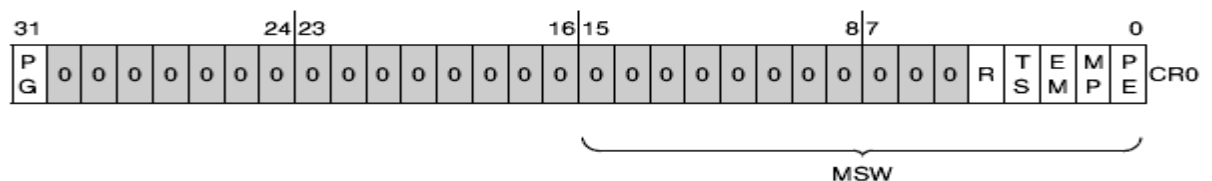
Selector (16bit)

Control Register:

Three Control registers CR0,CR2 & CR3.

(CR1 is left undefined by Intel).

1) CR0:



PE(Protection Enable):

PE=1 Protected mode

PE-0 Real mode

MP (Math processor):

If math coprocessor is present

EM(Emulation):

EM=1 emulate coprocessor

TS: Task switched

ET: Extension Type

ET=1 ; 80387 connected

ET=0; 80287 connected

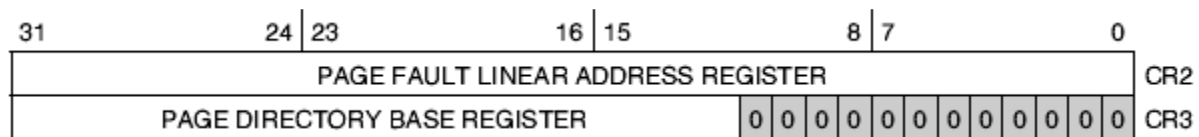
PG=Page Enable

If PG=1 paging enabled

i.e. memory is divided into fixed size pages of 4KB

PG=0 paging disabled

CR2 is read only which gives the last 32 bit address that caused page fault.



Debug Register:

31	16 15															0													
BREAKPOINT 0 LINEAR ADDRESS																	DR0												
BREAKPOINT 1 LINEAR ADDRESS																	DR1												
BREAKPOINT 2 LINEAR ADDRESS																	DR2												
BREAKPOINT 3 LINEAR ADDRESS																	DR3												
Int _E reserved. Do not define.																	DR4												
Int _E reserved. Do not define.																	DR5												
0												B	B	B	0	0	0	0	0	0	0	0	0	0	B	B	B	B	DR6
LEN	R	W	LEN	R	W	LEN	R	W	LEN	R	W	0	0	G	0	0	G	L	G	L	G	L	G	L	G	L	G	L	DR7
3	3	3	2	2	2	1	1	1	0	0	0	0	0	D	0	0	E	E	3	3	2	2	1	1	0	0	0	0	
31	16 15															0													

NOTE: 0 indicates Intel reserved: Do not define; SEE SECTION 2.3.10

These registers hold up to four linear address breakpoints.

The addresses in registers are compared with processors address generation logic on every instruction and if match is found an exception 1 (debug fault) is generated.

The debug address registers are effective whether or not paging is enabled.

Test Registers

Two Registers TR6 and TR7:

31											12	11							0					
LINEAR ADDRESS												V	D	D #	U	U #	W	W #	0	0	0	0	C	TR6
PHYSICAL ADDRESS												0	0	0	0	0	0	0	P L	REP		0	0	TR7

NOTE: 0 indicates Intel reserved: Do not define; SEE SECTION 2.3.10

-The test registers are used to perform confidence checking on the paging MMU's translation look aside buffer (TLB).

-By writing into this register one can initiate write directly into 80386 TLB or perform a mock TLB lookup.

-TR6 is test command register and TR7 is test data register.

Let's check the take away from this lecture

3) Virtual Mode Flag bit can be set using _____instruction or any task switch operation only

in the _____ mode

a) IRET, Virtual b) POPF, Real c) IRET, protected_ d) POPF, protected

4) The _____ bit decides whether it is a system descriptor or code/data segment descriptor

a) P b) S c) D d) G

L22. Exercise

Q.4 Draw the programmers model of Intel 80386Dx.

Q.5 Draw the structure of debug register.

Questions/Problems for practice:

Q.6 Draw and explain control register structure.

Learning from this lecture ‘Programmers model’:

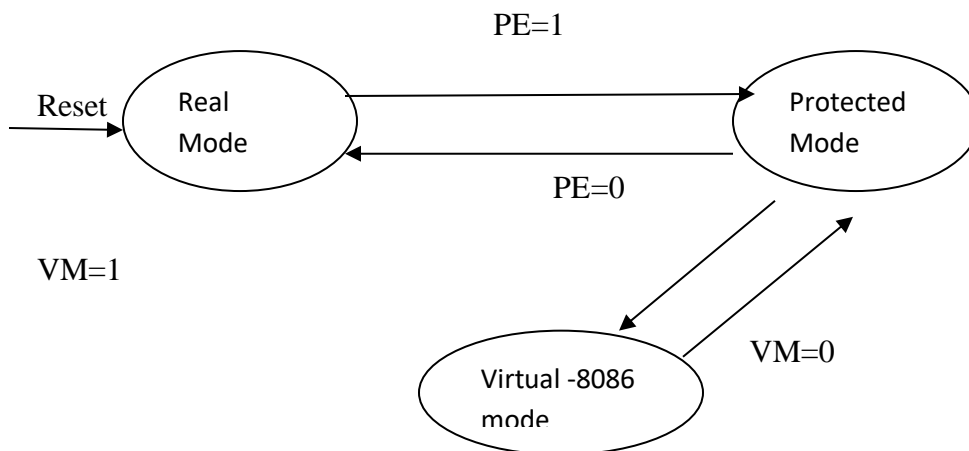
Students will able to know register set of Intel 80386Dx for programming purpose.

Lecture: 27

Operating modes

Learning Objective: In this lecture students will able to describe the different operating modes of Intel 80386DX.

4.9.4 Operating Modes Transition diagram:



a) Real Mode:

After reset, the 80386 starts from memory location FFFFFFF0H under the real address mode. In the real mode, 80386 works as a fast 8086 with 32-bit registers and data types.

-In real mode, the default operand size is 16 bit but 32-bit operands and addressing modes may be used with the help of override prefixes.

-The segment size in real mode is 64k, hence the 32-bit effective addressing must be less than 0000FFFFH.

-The real mode initializes the 80386 and prepares it for protected mode

-Memory Addressing in Real Mode: In the real mode, the 80386 can address at the most 1Mbytes of physical memory using address lines A₀-A₁₉.

-Paging unit is disabled in real addressing mode, and hence the real addresses are the same as the physical addresses.

-To form a physical memory address, appropriate segment registers contents (16-bits) are shifted left by four positions and then added to the 16-bit offset address formed using one of the addressing modes, in the same way as in the 80386 real address mode.

-The segment in 80386 real mode can be read, write or executed, i.e. no protection is available.

-Any fetch or access past the end of the segment limit generate exception 13 in real address mode.

-The segments in 80386 real mode may be overlapped or non-overlapped.

-The interrupt vector table of 80386 has been allocated 1Kbyte space starting from 00000H to 003FF.

b) Protected mode:

When PE=1, processor enter into the protected mode.

-All the capabilities of 80386 are available for utilization in its protected mode of operation.

-The 80386 in protected mode support all the software written for 80286 and 8086 to be executed under the control of memory management and protection abilities of 80386.

-The protected mode allows the use of additional instruction, addressing modes and capabilities of 80386.

-support paging mechanism

c) Virtual 8086 mode:

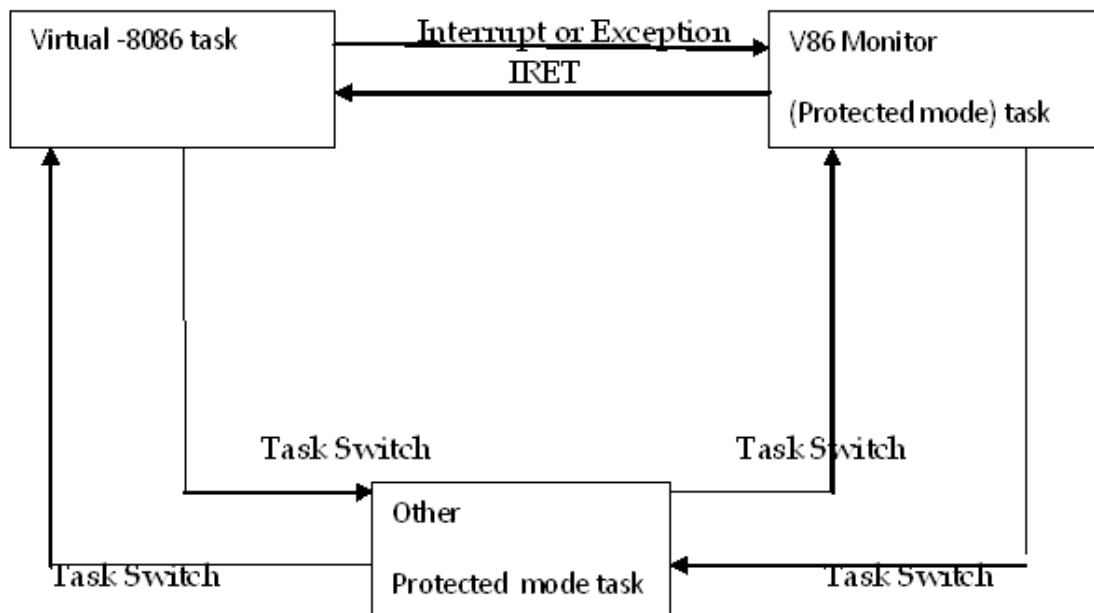
- In its protected mode of operation, 80386DX provides a virtual 8086 operating environment to execute the 8086 programs.
- The real mode can also be used to execute the 8086 programs along with the capabilities of 80386, like protection and a few additional instructions.
- Once the 80386 enters the protected mode from the real mode, it cannot return back to the real mode without a reset operation.
- Thus, the virtual 8086 mode of operation of 80386, offers an advantage of executing 8086 programs while in protected mode.
- The address forming mechanism in virtual 8086 mode is exactly identical with that of 8086 real mode.
- In virtual mode, 8086 can address 1Mbytes of physical memory that may be anywhere in the 4Gbytes address space of the protected mode of 80386.
- Like 80386 real mode, the addresses in virtual 8086 mode lie within 1Mbytes of memory.
- In virtual mode, the paging mechanism and protection capabilities are available at the service of the programmers.
- The 80386 supports multiprogramming, hence more than one programmer may be use the CPU at a time.
- Paging unit may not be necessarily enable in virtual mode, but may be needed to run the 8086 programs which require more than 1Mbytes of memory for memory management function.
- In virtual mode, the paging unit allows only 256 pages, each of 4Kbytes size.
- Each of the pages may be located anywhere in the maximum 4Gbytes physical memory. The virtual mode allows the multiprogramming of 8086 applications.
- The virtual 8086 mode executes all the programs at privilege level 3. Any of the other programs may deny access to the virtual mode programs or data.
- However, the real mode programs are executed at the highest privilege level, i.e. level 0.
- The virtual mode may be entered using an IRET instruction at CPL=0 or a task switch at any CPL, executing any task whose TSS is having a flag image with VM flag set to 1.
 - The IRET instruction may be used to set the VM flag and consequently enter the virtual mode.
 - The PUSHF and POPF instructions are unable to read or set the VM bit, as they do not access it.

-Even in the virtual mode, all the interrupts and exceptions are handled by the protected mode interrupt handler.

-To return to the protected mode from the virtual mode, any interrupt or execution may be used.

-As a part of interrupt service routine, the VM bit may be reset to zero to pull back the 80386 into protected mode.

-Entering & Leaving virtual 8086 mode:-



Entering The Virtual 8086 mode:

The processor can enter into Virtual 8086 mode by either of the two ways:

- 1) A Virtual 8086 task may be entered by a task switch, which loads a new 32-bit TSS that has the VM bit set in its copy of the EFLAG register. If the VM bit is not set, the 80386 executes the new task as a normal protected mode task.
- 2) The another way of entering the virtual 8086 mode is by the execution of an IRET instruction from a procedure whose CPL is 0.

Leaving the Virtual mode:

The processor leaves the virtual 8086 mode when interrupt or exception occur.

Let's check the take away from this lecture

5. --bit of CR0 register is used to enter into protected mode.

- i) PE ii) VM
- iii) IOPL iv) PG

6. ---register used as status register.

- i) CR0 ii) CR1 iii) CR2 iv) CR3

L23 Exercise:

Q.7 List the operating modes of Intel 80386Dx.

Q.8 Write short note on : Virtual 8086 mode

Questions/problems for practice:

Q.10 Draw the transition diagram of Operating modes of Intel 80386DX.

Q.11 When to enter and exit from Virtual mode.

Learning from the lecture 'Operating Modes':

Student will able to understand the different operating modes on Intel 80386Dx.

Lecture: 28

Address translation mechanism in protected mode : Segmentation

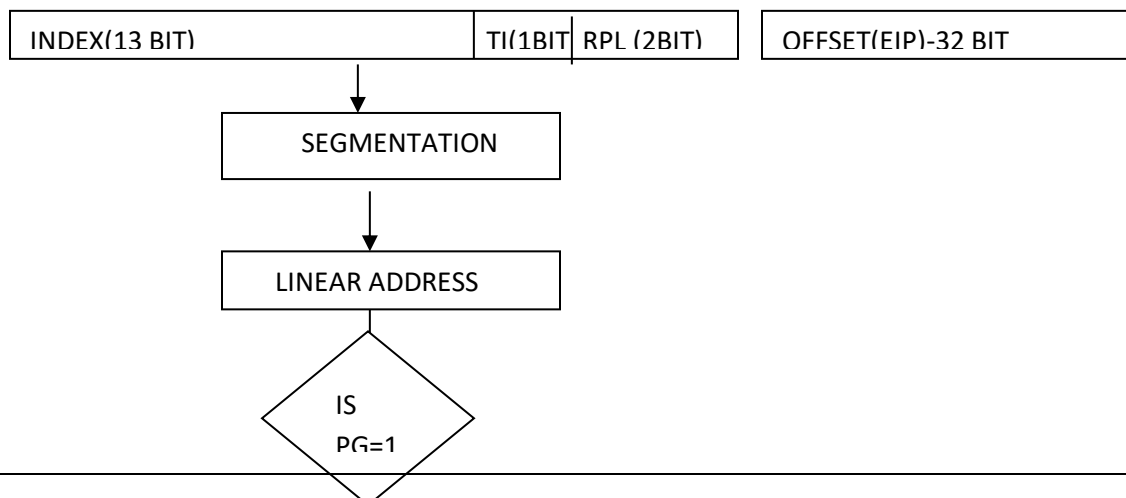
Learning Objective: In this lecture students will able to understand the need of address translation mechanism in protected mode.

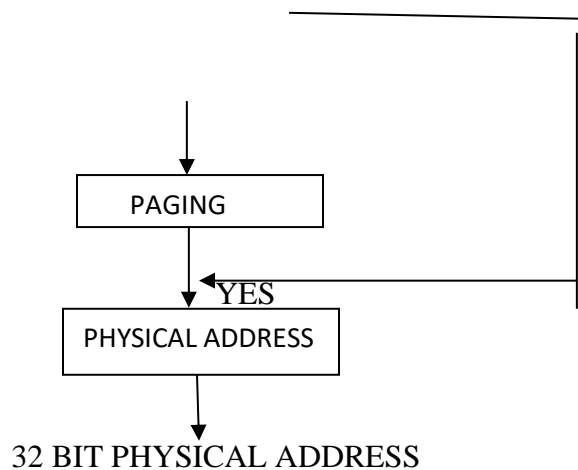
4.9.5 Protected mode address translation:

- 1) Address translation
- 2) Segmentation
- 3) Paging
- 4) TLB

a) Flowchart:

46 bit Virtual address (32 bit offset + 16 bit selector) + 2 bit for segmentation protection purpose





In protected mode CS does not give the base address. Here CS & Offset (32 bit) will give the virtual address.

b) Segmentation Mechanism (Logical to Linear address Translation):-

Description:

- 1) Logical address contain 32 bit offset part & 16 bit Selector part.
- 2) 13 bit Index part of selector is multiplied by 8 and used as a pointer to the desired descriptor in the descriptor table. As the length of each descriptor in descriptor table is 8 bytes; every time the index value is multiplied by 8 to select the desired descriptor.
- 3) The choice of the table to look for required descriptor will depend on TI- bit of selector.

For TI=0 – Global Descriptor Table (GDT)

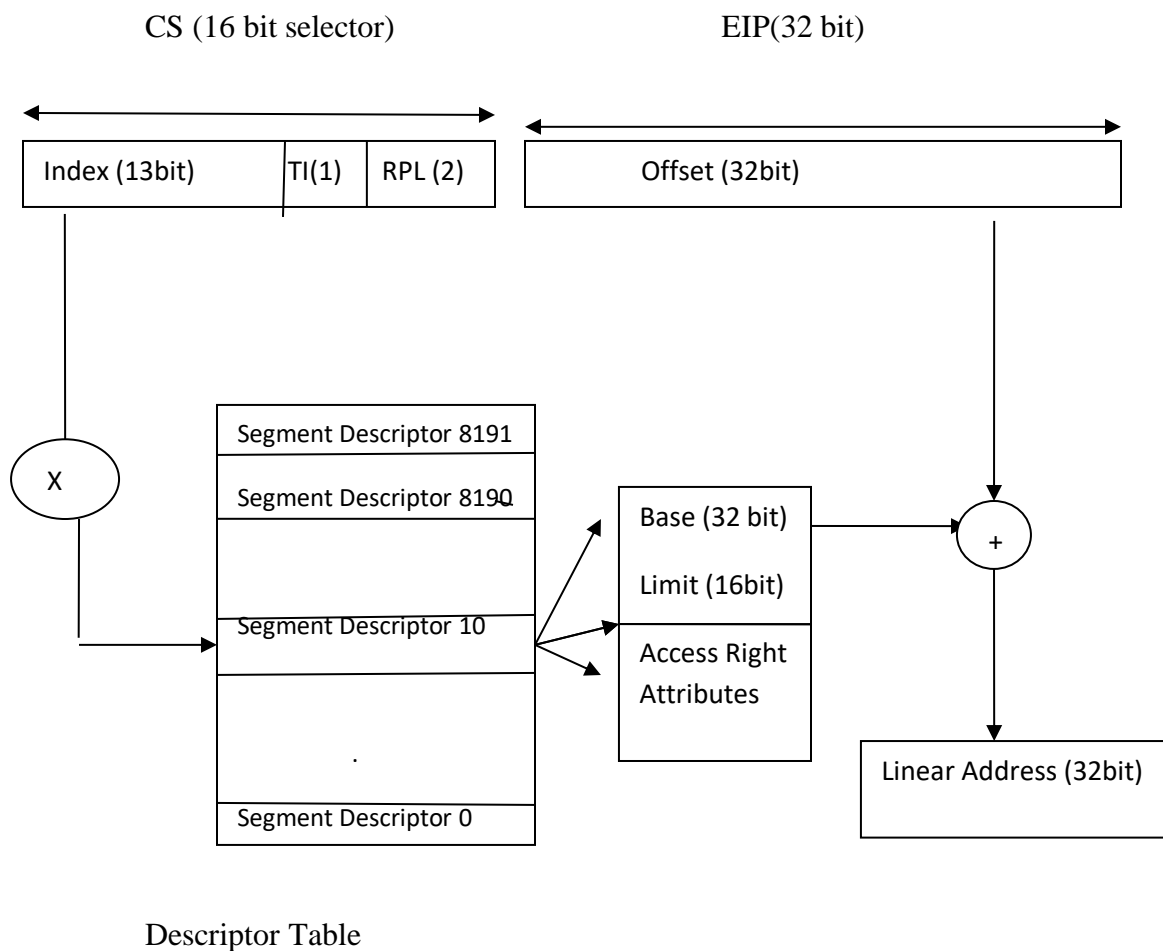
TI=1 – Local descriptor table (LDT)

- c) These tables present in memory are located by the use of GDTR or LDTR registers.
- d) The segment descriptor mainly contains the base address, limit & access control information applicable to that segment.
- e) 32bit base address from segment descriptor is added to the 32 bit offset part of logical address to obtain the 32-bit Linear address.

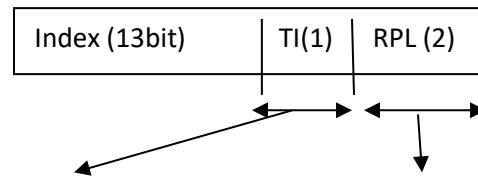
- f) If paging is disabled, the 32bit linear address obtained is the 32 bit physical address. However, if paging is enabled the paging hardware converts the linear address into physical address by using second stage of address translation.

To perform segmentation 4 types of data structures used:-

- 1) Selector
- 2) Segment Descriptor
- 3) Descriptor Table
- 4) Segment Register



1) Selector:-



Requested Privilege Level

Table Indicator	00 Highest PL
0 for GDT	01
1 for LDT	02
	03 Lowest PL

-16 bit segment registers CS,DS,SS,ES,FS,GHS are called segment selectors.

-16 bit selector as divided into three fields.

RPL(2 bits)- Requested Privilege Level

These two bits are used to represent four privilege levels PL0 to PL3.

These bits are used in protection checks to determine if access to segment is allowed or not instead of segment translation mechanism.

TI (1 bit)- Table Indicator

It is used to select one of the segment descriptor table.

If TI=0 ; GDT selected

TI=1: LDT selected

Index(13 bit):

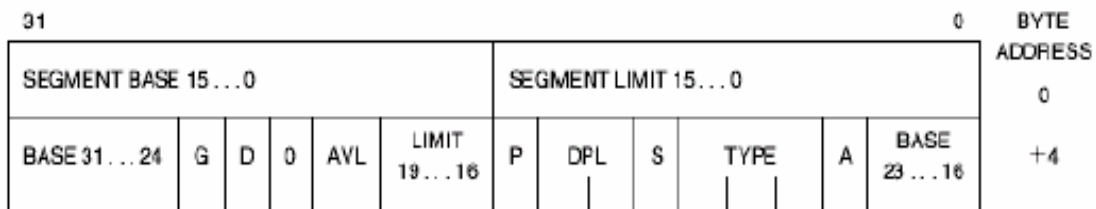
It is used to select one of the 8192 descriptors available in either GDT or LDT

2) Segment Descriptor:

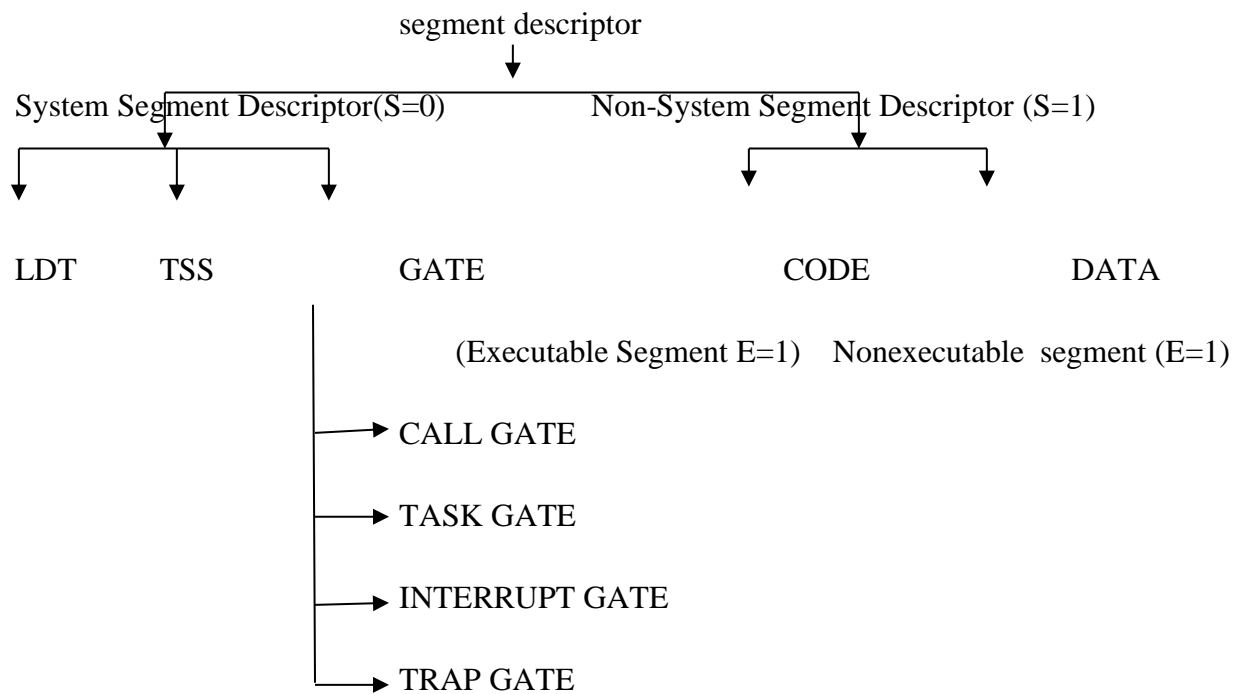
-A segment is described by a special structure called Segment descriptor

- Every segment has a descriptor
- It gives base address , size or limit of segment & attribute of segment.
- size of segment descriptor is 64 bits or 8 bytes
- Maximum no. of segment descriptor in LDT- 8192
- Maximum no. of segment descriptor in GDT- 8192

-Format of segment descriptor:



-Types of segment descriptor



Fields of Segment Descriptor:

Base address bits : 32

Limit Address : 20 bits (Length of segment –1)

A (Accessed) : Processor automatically sets this bit whenever a memory reference is made using the defined segment.

DPL : (2) : Indicates the level of privilege associated with the memory space that descriptor defines. DPL0 is highest

S (System) : If clear indicates that this is system segment descriptor. If set non-system.

P (Present): If clear, the address range that this descriptor defines is considered to be temporarily not present in physical memory space.

U (User): Undefined and ignored by 386, but user can use it.

X : Reserved

D (Default): When clear operands in this segment is assumed to be 16 bits and when set assumed to be 32 bits.

G (Granularity): When this bit is cleared, the 20 bit limit field is assumed to be measured in units of 1 byte. If set the limit is in units of 4 KB.

Type (3): Type of segment you are defining.

000: Data , read only (ROM space)

001: Data ,R/W

010: Stack read only

011: Stack R/W

100: Code, execute only

101: Code execute/ read

110: Code execute only , conforming

111: Code execute / read, conforming

3)Descriptor Table:

- Collection of all descriptors of similar type is called descriptor table
- All descriptor table stored in physical memory
- 1)Local Descriptor Table (LDT):-

Collection of all local descriptors of a particular task is called local descriptor table for that task.

There will be many LDT's depending upon no of tasks.

Maximum 8192 segment descriptors are present in the LDT, hence maximum size of LDT is 64 KB.

LDTR is a pointer/selector which points to the descriptor of the LDT of the task currently being executed.

- 2)Global Descriptor Table (GDT):

Collection of descriptors of global memory is called global descriptor table

There will be only one GDT in physical memory.

- Maximum 8192 segment descriptors are present in the GDT, hence maximum size of GDT is 64 KB.
- To access the GDT we need the base of GDT and size (Limit) . As GDT is variable length. This information is given in GDTR.

3)Interrupt Descriptor Table (IDT):

-In protected mode processor makes use of IDT & IDTR for dealing with the interrupts.

-the IDT contains 8 byte gate descriptors for task, trap or interrupt gates.

-the IDT contains maximum 256 descriptors

-IDT may present anywhere in the physical memory

4)Segment Register:

Let's check the take away from this lecture

5. Segmentation unit allows segments of _____size at maximum.

- a) 4Gbytes b) 6Mbytes c) 4Mbytes d) 6Gbytes

6. The _____bit decides whether it is a system descriptor or code/data segment descriptor

- a) P b) S c) D d) G

7. How many segment descriptor are available in GDT ?

- 1)8192 2)65536 3)16384

Q.7 List the operating modes of Intel 80386Dx.

Q.8 Write short note on : Virtual 8086 mode

Questions/problems for practice:

Q.10 Draw the transition diagram of Operating modes of Intel 80386DX.

Q.11 When rto enter and exit from Virtual mode.

Learning from the lecture ‘Operating Modes’:

Student will able to understand the different operating modes on Intel 80386Dx.

Lecture: 29

Address translation mechanism in protected mode : Paging

Learning Objective: In this lecture students will able to understand the need of address translation mechanism in protected mode.

4.9.6 Protected mode address translation:

- 1) Address translation
- 2) Segmentation
- 3) Paging
- 4) TLB

c) Paging Mechanism (Linear to Physical address translation):-

•PAGING OPERATION:

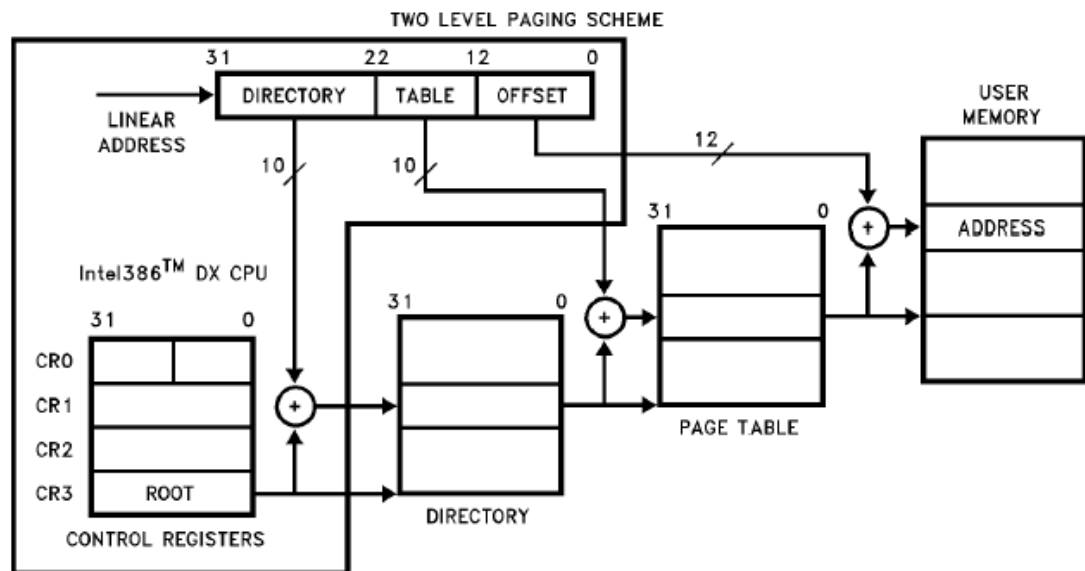
- The use of Paging feature is optional.
- Paging is one of the memory management techniques used for virtual memory multitasking operating system.
- The segmentation scheme may divide the physical memory into a variable size-segments but the paging divides the memory into a fixed size pages.
- The segments are supposed to be the logical segments of the program, but the pages do not have any logical relation with the program.
- The pages are just fixed size portions of the program module or data.
- The advantage of paging scheme is that the complete segment of a task need not be in the physical memory at any time.

- Only a few pages of the segments, which are required currently for the execution need to be available in the physical memory. Thus the memory requirement of the task is substantially reduced, relinquishing the available memory for other tasks.
- Whenever the other pages of task are required for execution, they may be fetched from the secondary storage.
- The previous page which are executed, need not be available in the memory, and hence the space occupied by them may be relinquished for other tasks.
- Thus paging mechanism provides an effective technique to manage the physical memory for multitasking systems.

Paging Unit:

- The paging unit of 80386 uses a two level table mechanism to convert a linear address provided by segmentation unit into physical addresses.
- The paging unit converts the complete map of a task into pages, each of size 4K. The task is further handled in terms of its page, rather than segments.
- The paging unit handles every task in terms of three components namely page directory, page tables and page itself.

Paging Mechanism:



•**Paging Descriptor Base Register:**

- The control register CR₂ is used to store the 32-bit linear address at which the previous page fault was detected.
- The CR₃ is used as page directory physical base address register, to store the physical starting address of the page directory.
- The lower 12 bit of the CR₃ are always zero to ensure the page size aligned directory. A move operation to CR₃ automatically loads the page table entry caches and a task switch operation, to load CR₀ suitably.

•**Page Directory :**

This is at the most 4Kbytes in size. Each directory entry is of 4 bytes, thus a total of 1024 entries are allowed in a directory.

- The upper 10 bits of the linear address are used as an index to the corresponding page directory entry. The page directory entries point to page tables.

•**Page Tables:** Each page table is of 4Kbytes in size and many contain a maximum of 1024 entries. The page table entries contain the starting address of the page and the statistical information about the page.

The upper 20 bit page frame address is combined with the lower 12 bit of the linear address. The address bits A₁₂- A₂₁ are used to select the 1024 page table entries. The page table can be shared between the tasks.

Page Directory Entry (PDE)

31	12	11	10	9	8	7	6	5	4	3	2	1	0
PAGE TABLE ADDRESS 31..12		OS RESERVED			0	0	D	A	0	0	U — S	R — W	P

Page Table Entry (PTE)

31	12	11	10	9	8	7	6	5	4	3	2	1	0
PAGE FRAME ADDRESS 31..12		OS RESERVED			0	0	D	A	0	0	U — S	R — W	P

U/S	R/W	Permitted Level 3	Permitted Access Levels 0, 1, or 2
0	0	None	Read/Write
0	1	None	Read/Write
1	0	Read-Only	Read/Write
1	1	Read/Write	Read/Write

- The P bit of the above entries indicate, if the entry can be used in address translation.
- If P=1, the entry can be used in address translation, otherwise it cannot be used.
- The P bit of the currently executed page is always high.
- The accessed bit A is set by 80386 before any access to the page. If A=1, the page is accessed, else unaccessed.
- The D bit (Dirty bit) is set before a write operation to the page is carried out. The D-bit is undefined for page director entries.
- The OS reserved bits are defined by the operating system software.
- The User / Supervisor (U/S) bit and read/write bit are used to provide protection. These bits are decoded to provide protection under the 4 level protection model.
- The level 0 is supposed to have the highest privilege, while the level 3 is supposed to have the least privilege.
- This protection provide by the paging unit is transparent to the segmentation unit.

Let's check the take away from this lecture

9.bit of CR0 is used to enabled the paging mechanism.

- | | |
|-----------|-----------------------|
| i) PG | ii) PE |
| iii) IOPL | iv) none of the above |

10. Size of Page table is---

- | | |
|-----------|----------|
| i) 4 KB | ii) 1 KB |
| iii) 32KB | iv) 4GB |

L24 Exercise:

Q.12 Define paging. Explain data structure used for paging.

Q.13 What is TLB. Draw the structure of TLB.

Question/ problems for practice

Q.15 Explain how to convert linear address to physical address.

Q.16 Explain paging mechanism.

Learning from the Lecture ‘Paging’:

Student will able to understand the need of address translation.

Lecture: 30**Protection mechanism**

Learning Objective: In this lecture students will able to understand the need of Protection mechanism in multitasking environment.

4.9.7 Need of protection:

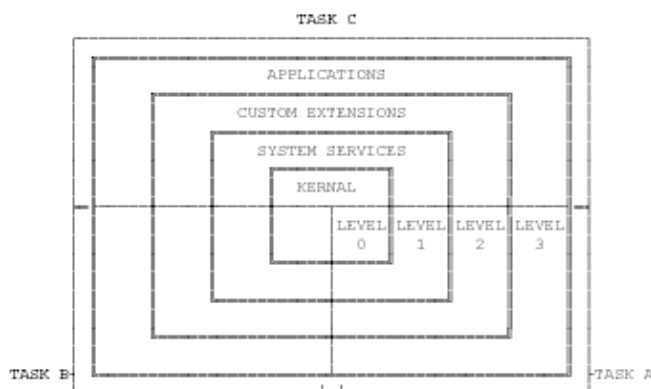
1) Individual task are required to be protected from each other.

2) Global resources are required to be protected from all the tasks.

It's one of the technique for implementing System level protection and supported directly by hardware.

The privilege system consists of four levels of privilege numbered from 0 to 3. (PL0 to PL3) being PL0 highest and PL3 lowest.

Protection model:



Processor supports two protection schemes:

1) Segment level protection

2) Page level protection

1) Segment Level Protection:

In order to prevent any unwanted access to memory, the processor performs following different checks:

1. Type checking
2. Limit checking
3. Addressable domain check
4. Procedure entry point check
5. Privileged instruction check

First 2 aspects of protection are performed by checking protection parameters within the segment descriptors such as limit, type & PL. Remaining 3 aspects of protection are implemented using PL. Any violation of these protection checks results in an exception.

1) Type checking:

Type field of the descriptor specifies type of the descriptor and the intended usages of segment.

e.g. On a read-only segment, memory write is not allowed.

2) Limit Checking:

The processor uses limit field of a segment descriptor to prevent programs from addressing outside the segments.

The granularity bit is used to decide how the limit field can be used.

3) Addressable Domain check:

A task can access segments with the same or lower PL. The DPL field of the descriptor is used to specify the PL of the segment.

e.g. A task with CPL of 0 may access data operands in segments with a any privilege level because CPL0 is the highest PL.

4) Procedure Entry point check:

- 80386 performs the procedure entry point check by the use of a special descriptor known as call gate.

-Call gate are used to control the transfer of execution between procedures of different privilege levels.

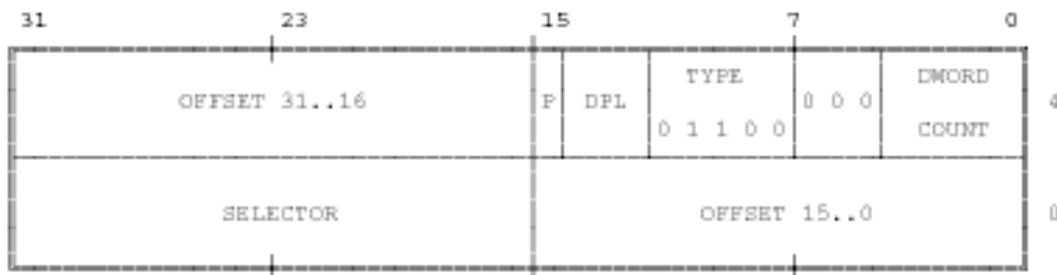
-It is only mechanism that allows to call a procedure having higher privilege level.

-Call gates are used by JMP and CALL instructions and may only reside in GDT or LDT just like segment or other descriptors. It provides the protected indirect access to the procedure at different privilege level.

-The call gate descriptor contains two important things:

1. Selector which point to the descriptor for the segment where the procedure is actually loaded and
2. Offset of the called procedure in its segment. When a CALL is made through call gate the selector of the call gate is used to access the segment descriptor.

Processor uses the base address from the segment descriptor and the offset from the call gate descriptor to calculate the physical address of the called procedure.



Remaining bits are:

P : present bit

This bit is used to indicate the presence of segment in the memory.

DPL: Descriptor Privilege Level

DWORD Count:

This field is used to specify, the number of double words to transfer from the caller's stack to the called procedures stack. If there is a change of PL.

5) Privileged Instructions:

All the privileged instructions in must be executed only when the CPL is 0.

e.g. CLTS, LGDT , HLT ,LMSW etc.

2) Page Level Protection:

- 1) Restriction of addressable domain
- 2) Type checking

The U/S & R/W fields of page directory entry & page table entry are used to control access to pages.

Let's check the take away from this lecture

11. Which segment descriptor used for protection mechanism.

- i) Call Gate ii) Task Gate iii) Trap Gate iv) none of these

12. What is the size of segment descriptor

- i) 64 bytes ii) 32 bytes
iii) 8 bytes iv) none of the above

Exercise:

Q.18 Write the need of protection mechanism.

Q.19 Draw the structure of call gate descriptor.

Questions/problems for Practice

Q.19 Which mechanism is used to transfer control from lower privilege level to upper privilege level.

Learning from the lecture ‘Protection mechanism’: Student will be able to understand the need and what mechanism used for protection mechanism in multitasking environment.

Add to Knowledge (Content Beyond Syllabus)

Task Switching :The 80386 switches execution to another task in any of four cases:

1. The current task executes a JMP or CALL that refers to a TSS descriptor.
2. The current task executes a JMP or CALL that refers to a task gate.
3. An interrupt or exception vectors to a task gate in the IDT.
4. The current task executes an IRET when the NT flag is set.

JMP, CALL, IRET, interrupts, and exceptions are all ordinary mechanisms of the 80386 that can be used in circumstances that do not require a task switch. Either the type of descriptor referenced or the NT (nested task) bit in the flag word distinguishes between the standard mechanism and the variant that causes a task switch. To cause a task switch, a JMP or CALL instruction can refer either to a TSS descriptor or to a task gate. The effect is the same in either case: the 80386 switches to the indicated task. An exception or interrupt causes a task switch when it vectors to a task gate in the IDT. If it vectors to an interrupt or trap gate in the IDT, a task switch does not occur. Whether invoked as a task or as a procedure of the interrupted task, an interrupt handler always returns control to the interrupted procedure in the interrupted task. If the NT flag is set, however, the handler is an interrupt task, and the IRET switches back to the interrupted task.

A task switching operation involves these steps:

1. Checking that the current task is allowed to switch to the designated task. Data-access privilege rules apply in the case of JMP or CALL instructions. The DPL of the TSS descriptor or task gate must be numerically greater (e.g., lower privilege level) than or equal to the maximum of CPL and the RPL of the gate selector. Exceptions, interrupts, and IRET are permitted to switch tasks regardless of the DPL of the target task gate or TSS descriptor.
2. Checking that the TSS descriptor of the new task is marked present and has a valid limit. Any errors up to this point occur in the context of the outgoing task. Errors are restartable and can be handled in a way that is transparent to applications procedures.
3. Saving the state of the current task. The processor finds the base address of the current TSS cached in the task register. It copies the registers into the current TSS (EAX, ECX, EDX, EBX, ESP, EBP, ESI, EDI, ES, CS, SS, DS, FS, GS, and the flag register). The EIP field of the TSS points to the instruction after the one that caused the task switch.
4. Loading the task register with the selector of the incoming task's TSS descriptor, marking the incoming task's TSS descriptor as busy, and setting the TS (task switched) bit of the MSW. The selector is either the operand of a control transfer instruction or is taken from a task gate.
5. Loading the incoming task's state from its TSS and resuming execution. The registers loaded are the LDT register; the flag register; the general registers EIP, EAX, ECX, EDX, EBX, ESP, EBP, ESI, EDI; the segment registers ES, CS, SS, DS, FS, and GS; and PDBR. Any errors detected in this step occur in the context of the incoming task. To an exception handler, it appears that the first instruction of the new task has not yet executed.

The state of the outgoing task is always saved when a task switch occurs. If execution of that task is resumed, it starts after the instruction that caused the task switch. The registers are restored to the values they held when the task stopped executing. Every task switch sets the TS (task switched) bit in the MSW (machine status word). The TS flag is useful to systems software when a coprocessor (such as a numerics

coprocessor) is present. The TS bit signals that the context of the coprocessor may not correspond to the current 80386 task. Exception handlers that field task-switch exceptions in the incoming task should be cautious about taking any action that might load the selector that caused the exception. Such an action will probably cause another exception, unless the exception handler first examines the selector and fixes any potential problem. The privilege level at which execution resumes in the incoming task is neither restricted nor affected by the privilege level at which the outgoing task was executing. Because the tasks are isolated by their separate address spaces and TSSs and because privilege rules can be used to prevent improper access to a TSS.

4.10 Learning Outcomes:

1. Know:

- a) Student should be able to differentiate between Intel 80386Dx and 80386SX architecture.
- b) List the advanced features of Intel 80386DX

2. Comprehend:

- a) Student should be able to Describe the advanced features such as address translation mechanism, paging, operating modes etc.
- b) Explain the need of protection mechanism in multitasking environment.

3. Apply, analyze and synthesize:

Student should be able to show the demonstration how to switch from real mode to protected mode using DPMI driver.

4.11. Viva Questions

Q.1 Differentiate between Intel 80386Dx and SX processor.

Ans: 80386DX - this CPU could work with 16-bit and 32-bit external buses.

80386SX - low cost version of the 80386. This processor had 16 bit external data bus and 24-bit external address bus.

Q.2 List the features of Intel 80386Dx.

Q.3 Which are the different operating modes.

Ans: Real mode, protected mode and virtual mode.

Q.4 Draw the flowchart of Address translation.

Q.5 Draw the structure of Status word.

4.12 Multiple Choice Questions.

1. The 80386 DX is.....bit microprocessor.

(a) 16 (b) 24 (c) 8 (d) 32
2. How much physical memory can be accessed by 80386DX microprocessor? (a) 1 MB (b) 4 MB (c) 4 GB (d) 64 TB

3. How much virtual memory can be accessed by 80386 microprocessor?
(a) 1 MB (b) 4 MB (c) 4 GB (d) 64 TB
4. The phenomenon of VIRTUAL MEMORY uses the concept of _____.
a). composing memories b). paging c). both d). none
5. The 80386 DX microprocessor has.....
(a) Pipelined architecture (b) separate data bus and address bus (c) Multitasking (d) all of above
6. What is the size of prefetch queue in 80386DX?
(a) 6 bytes (b) 10 bytes (c) 12 bytes (d) 16 bytes
7. What is the function of bus interface in 80386DX?
(a) To communicate between different parts of 80386DX (b) To generate control signals (c) To interface between 80386DX and input/output devices and memory (d) All of above
8. What is the size of barrel shifter in 80386DX?
(a) 32-bit (b) 48-bit (c) 64-bit (d) 80-bit
9. If PE = 1, then 80386 DX microprocessor operates in.....
(a) Real mode (b) Protected mode (c) Virtual 86 mode (d) Special mode
10. What happens after providing reset to 80386DX microprocessor?
(a) It starts execution from address FFFFFFF0H (b) ESP = FFFFFFFFH (c) 80386 operates in real mode (d) All of above
11. Which flag is not present in 8086 microprocessor but present in 80386DX microprocessor?
(a) Nested Task Flag (b) Sign Flag (c) Trap Flag (d) Zero Flag
12. Which IOPL value has highest priority?
(a) 00 (b) 02 (c) 01 (d) 03
13. How many segment registers are present in 80386?

(a) 4 (b) 2 (c) 6 (d) 8

14. How many break point addresses we can load in debug registers of 80386DX microprocessor?

(a) 4 (b) 5 (c) 7 (d) 8

15. What is the size of the LDTR?

(a) 16 bits (b) 32 bits (c) 48 bits (d) 64 bits

4.13 Short Answer questions

. Q.1 Differentiate between real mode and protected mode of X86 family.

Q 2. Explain control registers of 80386DX.

Q.3 what is a selector? What is its purpose?

Q.4 Explain how a linear address is generated using selectors and Descriptors.

Q.5 What is a page fault? When page fault occur?

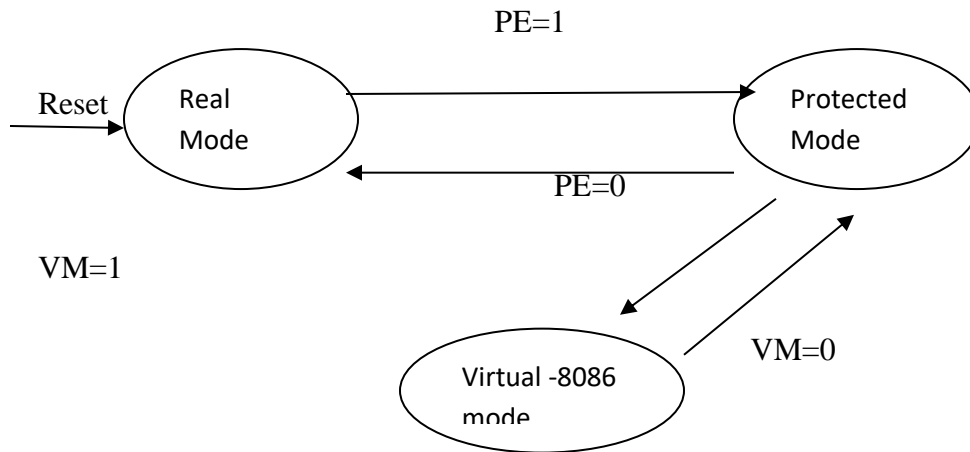
Q.6 How to switch from real mode to protected mode & protected to real?

Q. 7 Draw the structure of EFLAG register.

4.14 Long questions.

Q. 1. Draw the mode transition diagram of x86 processor & compare real & protected mode with respect to segment size, number of segments, page size, virtual memory support, addressing mechanism & interrupt processing.

Ans: Transition diagram:



Comparison:

	Real Mode	Protected mode
Segment size	64 KB	Min- 64 Kb Max -4 GB
Number of Segments	4	6
page size	Paging disable	4KB
virtual memory support	No	Yes
addressing mechanism	Segment translation only. Offset + segment (X10)= 20 bit physical address	Segmentation + paging
Interrupt processing	IVT	IDT
Entry in mode	On reset	When PE=0

Q.2 Explain segment translation mechanism of X86 processor with flowchart. Also explain segment descriptor fields.

Q.3 How the memory management is done in 80386DX?

Q.4 Differentiate between real mode and protected mode of X86 family.

Q.5 Draw protected mode address translation in X86 processor with neat flow diagram. Explain

segment translation in detail. List and explain fields of non-system segment descriptor.

Q.6 What is difference between GDT & LDT. Explain the role of GDTR, LDTR and IDTR.

Q.7 State the version of 80386 processor. Draw and explain the block diagram of

Intel 80386 processor.

Q.8 Show the register model of X86 processor & explain all control registers & memory management registers in detail.

Q.9 Explain segment translation mechanism of X86 processor with flowchart. Also explain segment descriptor fields. (may 2017)

Q. 10 a)Segment Translation: Logical to Linear address translation

b)Draw general segment descriptor format

Q.11 Explain EFLAG registers of 80386DX processor.

4.15. References

Advanced Microprocessors by Daniel Tabak

Intel Microprocessors: By Berry B. Brey

Advanced Microprocessor: By Roy & Bhurchandi

Practice for Module No.4 Intel 80386DX(based on University Patterns)

Q.1 A) Explain the block diagram of Intel 80386DX processor (10marks).

Q.2 A) Draw and explain the structure of Control Register. (10Marks)

B) Draw and explain the structure of EFLAG Register. (5 marks)

C) Draw and explain the structure of Debug register. (10marks)

D) Draw and explain the structure of System management registers. (10 marks)

Q.3 A) Explain how the memory management done in Intel 80386DX. (10marks)

B) Explain segment translation mechanism with flowchart. (10marks)

C) Explain paging mechanism and benefit of TLB. (10marks)

D) Differentiate between segmentation and paging mechanism (5 marks).

E) Differentiate between GDT, LDT and IDT. (5 marks)

Q.4 A) Explain in brief the protection mechanism? (10 marks)

B) What kind of mechanism used to transfer the control from lower privilege level to upper privilege level. (10marks)

C) Draw the structure of call gate descriptor. (5 marks)

Self-Assessment

- Q.1** Explain the features of Intel 80386DX processor and compare it with its predecessors.
Q.2 Write the steps to switch from real mode to protected mode .
Q.3 Explain how to enter and exit from virtual mode.
Q.4 Draw the structure of IDT.
Q.5 Explain how to compute size virtual memory.

Self-Evaluation

Name of Student		
Class		
Roll No.		
Subject		
Module No.		
S.No		Tick Your choice
16.	Do you understand the working of Intel 80386DX ?	<input type="radio"/> Yes <input type="radio"/> No
17.	Do you understand the difference between Intel 80386Dx and 80386SX?	<input type="radio"/> Yes <input type="radio"/> No
18.	Do you able to draw the structure of register set?	<input type="radio"/> Yes <input type="radio"/> No
19.	Do you understand the need of protection mechanism in multitasking environment?	<input type="radio"/> Yes <input type="radio"/> No
20.	Do you understand the logical, linear and physical address?	<input type="radio"/> Yes <input type="radio"/> No
21.	Do you understand module ?	<input type="radio"/> Yes, Completely. <input type="radio"/> Partialy. <input type="radio"/> No, Not at all.

Module: 5

Pentium Processor

5.1. Motivation:

At the completion of the subject, students should be able to:

- To understand features of Pentium-5.
- Advances in Pentium Architecture.
- Concept of Micro architecture
- To understand Intel Pentium Net-bus architecture.
- Compare Pentium family and multicore architecture

5.2. Syllabus:

Lec ture	Content	Duration	Self Study
32	Superscalar architecture Pentium Processor: Block Diagram	2 Lecture	2 hours
33	Superscalar Operation	1 Lecture	2 hours
34	Super pipelining : Integer & Floating Point Pipeline Stages	1 Lecture	2 hours
35	Data flow architecture	1 Lecture	2 hours
36	Comparison of Pentium 2, Pentium 3 and Pentium 4 Processors	1 Lecture	2 hours

5.3. Weightage: 15 Marks

5.4. Learning Objectives: Students should be able to-

1. **List** the advanced features of Pentium-1 such as DIB, superscalar architecture, Cache system , Branch prediction logic etc. and also relate P-1 with Intel 80386Dx, 80486Dx etc. **(R)**
2. **Describe** the superscalar architecture (More than one ALU) and super pipeline operations and analyze the performance of processor. **(U)**
3. **Distinguish** between different Pentium family processors w.r.t. different parameters. **(U)**

4. **Describe** the features of multicore processors like i3, i5 and i7.(R)
5. **Predict** how flushing / discarding of instructions from large size queue problem is solve using Branch prediction logic.(U)

6. **Recall** cache subsystem and understand the concept of MESI protocol in code and data cache organization.(R)

4.5. Theoretical Background:

Pentium is a brand used for a series of x86 architecture-compatible microprocessors produced by Intel since 1993. In their form as of November 2011, Pentium processors are considered entry-level products that Intel rates as "two stars", meaning that they are above the low-end Atom and Celeron series, but below the faster Core i3, i5, i7 and high-end Xeon series.

As of 2017, Pentium processors have little more than their name in common with earlier Pentiums, and are based on both the architecture used in Atom and that of Intel Core processors. In the case of Atom architectures, Pentiums are the highest performance implementations of the architecture. With Core architectures, Pentiums are distinguished from the faster, higher-end i-series processors by lower clock rates and disabling some features, such as hyper-threading, virtualization and sometimes L3 cache.

The name Pentium is originally derived from the Greek word *penta* (πεντα), meaning "five", a reference to the prior numeric naming convention of Intel's 80n86 processors (80186–80486), with the Latin ending *-ium*.

5.6. Abbreviations:

- SM Bus: System Management Bus
- BIU: Bus Interface Unit
- EU: Execution Unit
- ID: Instruction Decoder
- BTS: Branch Target Buffer
- TLB: Translation Look aside Buffer
- FPU: Floating Point Unit
- ALU: Arithmetic & Logic Unit
- MMU: Memory Management Unit
- SMM: System Management Mode
- BIST: Built In Self Test

- MMX: Multimedia Extension
- AGU: Address Generation Unit
- IEU : Integer Execution Unit
- SSE : Intel's Streaming SIMD Extensions Executions Unit
- BHB : Branch History Buffer

5.7. Formulae:

5.8. Key Definitions:

SMBUS: The System Management Bus (SMBus) is a two-wire interface through which simple power-related chips can communicate with rest of the system. It uses I2C as its backbone. With the SMBus, a device can provide manufacturer information, tell the system what its model or part number is, save its state for a suspend event, report different types of errors, accept control parameters and return its status.

INTEL MICROPROCESSORS: Microprocessors made by Intel Corporation form the foundation of all PCs. Models after the 8086 are often referred to by the last three digits (for example, the 286, 386, and 486). Many of the microprocessors come in different varieties that run at various clock rates. The 80486 architecture, for example, supports clock rates of from 33 to 66 MHz.

Superscalar architecture: More than one ALU.

Dynamic execution:

Speculative execution and out-of-order completion.

5.9. Course Content:

Lecture: 32 & 33

Pentium Processor: Block Diagram

Learning Objective: In this lecture student will able to learn hardware architecture.

5.9.1 Block Diagram: First introduced in 1993, the Pentium was the successor to Intel's 486 line of CPUs and the defining processor of the fifth generation. The original Pentium microprocessor had the internal code name P5, and was a pipelined in-order superscalar microprocessor, produced using a 0.8 μm process.

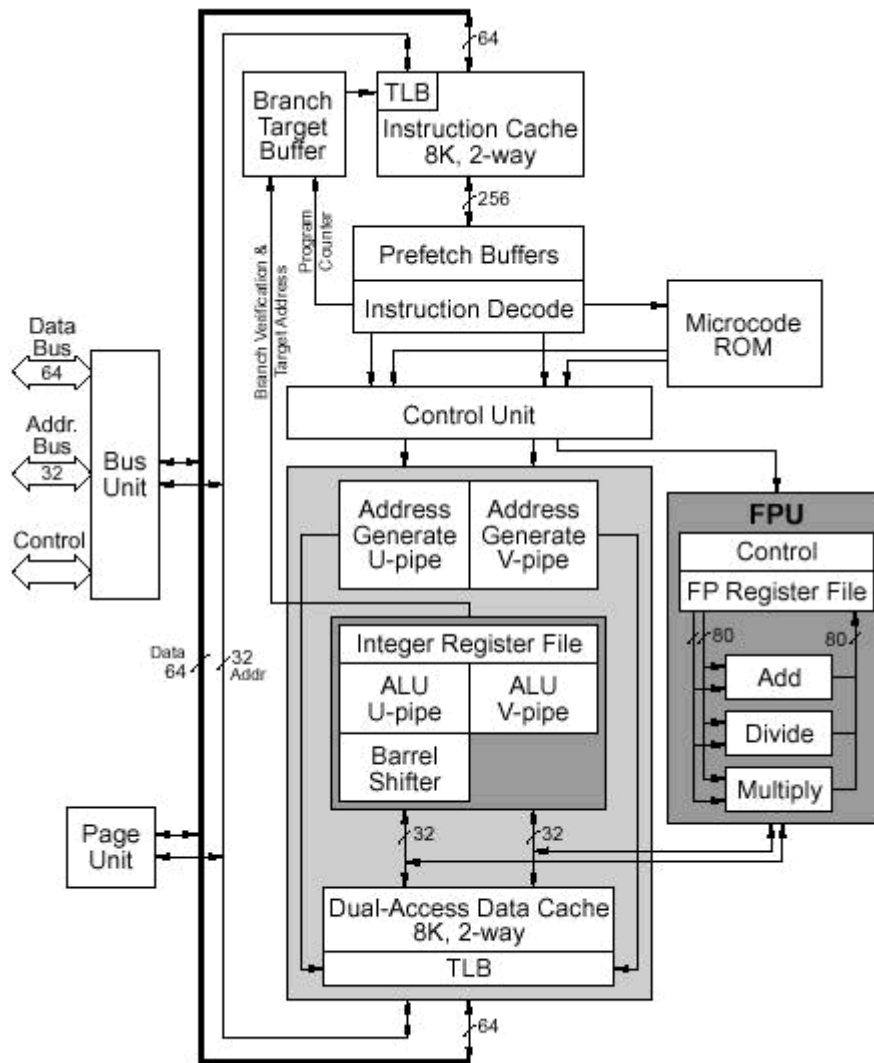


Figure 1. Pentium block diagram.

It was followed by the P54, a shrink of the P5 to a 0.6 μm process, which was dual-processor ready and had an internal clock speed different from the front side bus (it's much more difficult to increase the bus speed than to increase the internal clock). In turn, the P54 was followed by the P54C, which used a 0.35 μm process - a pure CMOS process, as opposed to the Bipolar CMOS process that was used for the earlier Pentiums. The subsequent P55C Pentium MMX processor was based on the P5 core and used the 0.35 μm fabrication process. It offered further significant improvements by doubling the size of the on-board primary cache to 32KB and by an extension to the instruction set to optimize the handling of multimedia functions.

The principal architectural changes that explained the Pentium's greatly increased performance over the 486 chips.

5.9.2 Instruction pairing rules of 'U' & 'V' pipeline in Pentium processor.

To understand the superscalar operation of u and v pipeline, we have to distinguish between simple and complex instructions.

- Simple instructions are entirely hardwired, do not require any microcode control and, in general, executes in one clock cycle
- The exceptions are the ALU mem,reg and ALU reg,mem instructions which are 3 and 2 clock operations respectively.
- Sequencing hardware is used to allow them to function as simple instructions.
- The following integer instructions are considered simple and may be paired:
 1. mov reg, reg/mem/imm
 2. mov mem, reg/imm
 3. alu reg, reg/mem/imm
 4. alu mem, reg/imm
 5. inc reg/mem
 6. dec reg/mem
 7. push reg/mem
 8. pop reg
 9. lea reg,mem

- 10. jmp/call/jcc near
- 11. nop
- 12. test reg, reg/mem
- 13. test acc, imm

- In order to issue two instructions simultaneously they must satisfy the following conditions:
 - Both instructions in the pair must be “simple”.
 - There must be no read-after-write or write-after-write register dependencies between them
 - Neither instruction may contain both a displacement and an immediate
 - Instructions with prefixes can only occur in the u-pipe

Let's check the take away from this lecture

- 1) Size of data bus in Pentium 1.
 - i) 32 bits
 - ii) 64 bits
 - iii) 16 bits
 - iv) none of the above
- 2) Number of ALU's
 - i) 2
 - ii) 1
 - iii) 3
 - iv) 4

L21. Exercise:

Q.1 List the features of Pentium1.

Q.2 Write instruction issue algorithm for Pentium 1.

Questions/problems for practice:

Q. 3 Write rules for pairing of instructions.

Learning from the lecture ‘Block Diagram’:

Student will able to explain the hardware architecture of Pentium1.

Lecture: 34

Superscalar Operation

Learning objective: In this lecture students will able to understand superscalar and superpipelining architecture of Pentium 1.

9.3 Superscalar Architecture: The Pentium family of processors originated from the 80486 microprocessor. The term "Pentium processor" refers to a family of microprocessors that share a common architecture and instruction set. The first Pentium processors were introduced in 1993. It runs at a clock frequency of either 60 or 66 MHz and has 3.1 million transistors. Some of the features of Pentium architecture are

- Complex Instruction Set Computer (CISC) architecture with Reduced Instruction Set Computer (RISC) performance.
- 64-Bit Bus
- Upward code compatibility.
- Pentium processor uses Superscalar architecture and hence can issue multiple instructions per cycle.
- Multiple Instruction Issue (MII) capability.
- Pentium processor executes instructions in five stages. This staging, or pipelining, allows the processor to overlap multiple instructions so that it takes less time to execute two instructions in a row.
- The Pentium processor fetches the branch target instruction before it executes the branch instruction.
- The Pentium processor has two separate 8-kilobyte (KB) caches on chip, one for instructions and one for data. It allows the Pentium processor to fetch data and instructions from the cache simultaneously.
- When data is modified, only the data in the cache is changed. Memory data is changed only when the Pentium processor replaces the modified data in the cache with a different set of data
- The Pentium processor has been optimized to run critical instructions in fewer clock cycles than the 80486 processor.

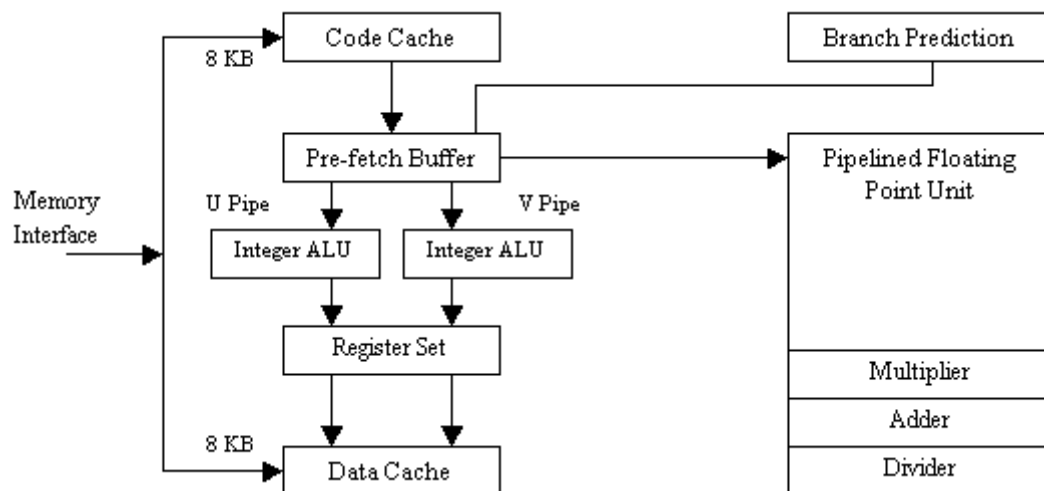


Fig. Superscalar Architecture of Pentium

The Pentium processor has two primary operating modes -

1. **Protected Mode** - In this mode all instructions and architectural features are available, providing the highest performance and capability. This is the recommended mode that all new applications and operating systems should target.
2. **Real-Address Mode** - This mode provides the programming environment of the Intel 8086 processor, with a few extensions. Reset initialization places the processor in real mode where, with a single instruction, it can switch to protected mode

Superpipelining:

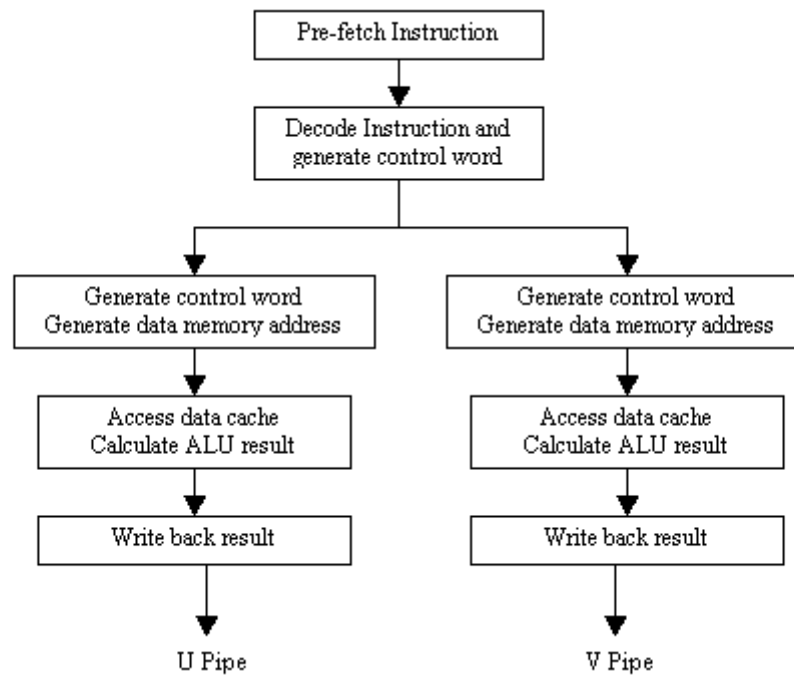


Fig Pentium pipeline stages

Let's check the take away from this lecture

3) No. of Integer pipeline stages

- | | |
|--------|-------|
| i) 5 | ii) 7 |
| iii) 6 | iv) 8 |

4) No. of ALU

- | | |
|--------|-------|
| i) 1 | ii) 2 |
| iii) 3 | iv) 4 |

L22. Exercise

Q.4 What is superscalar architecture.

Q.5 Explain superpipelining architecture..

Questions/Problems for practice:

Q.6 Explain Pentium pipeline stages.

Learning from this lecture 'Superscalar':

Students will able to understand the superscalar and superpipelining architecture.

Lecture: 35

Integer & Floating Point Pipeline Stages

Learning Objective: In this lecture students will be able to list the different pipeline stages..

9.5 Integer pipeline stages:

The Pentium's basic integer pipeline is five stages long, with the stages broken down as follows:

1. **Pre-fetch/Fetch** : Instructions are fetched from the instruction cache and aligned in pre-fetch buffers for decoding.
2. **Decode1** : Instructions are decoded into the Pentium's internal instruction format. Branch prediction also takes place at this stage.
3. **Decode2** : Same as above, and microcode ROM kicks in here, if necessary. Also, address computations take place at this stage.
4. **Execute** : The integer hardware executes the instruction.
5. **Write-back** : The results of the computation are written back to the register file.

9.6 The floating-point unit (FPU) of the Pentium processor is heavily pipelined.

- The FPU is designed to be able to accept one floating-point operation every clock.

It can receive up to two floating-point instructions every clock, one of which must be an exchange instruction (FXCH).

- The 8 FP pipeline stages are summarized below:

1. **PF Prefetch**
2. **D1 Instruction Decode**
3. **D2 Address generation**
4. **EX Memory and register read:** This stage performs register reads or memory reads required
5. **X1 Floating-Point Execute stage one:** conversion of external memory format to internal FP data
6. **X2 Floating-Point Execute stage two**

7. **WF** Perform rounding and write floating-point result to register file

8. **ER** Error Reporting/Update Status Word.

- The rules of how floating-point (FP) instructions get issued on the Pentium processor are:

1. FP instructions do not get paired with integer instructions.

2. When a pair of FP instructions is issued to the FPU, only the FXCH instruction can be the second instruction of the pair.

The first instruction of the pair must be one of a set F where $F = [\text{FLD}, \text{FADD}, \text{FSUB}, \text{FMUL}, \text{FDIV}, \text{FCOM}, \text{FUCOM}, \text{FTST}, \text{FABS}, \text{FCHS}]$.

1. FP instructions other than FXCH and instructions belonging to set F, always get issued singly to the FPU.

FP instructions that are not directly followed by an FP exchange instruction are issued singly to the FPU

Let's check the take away from this lecture

5. No. of floating point pipeline stages

i) 8

ii) 4

iii) 5

iv) 10

6. No. of FPU

i) 1

ii) 2

iii) 3

iv) none of the above

L23 Exercise:

Q.7 Define pipeline. List the integer pipeline stages.

Q.8 List the floating point pipeline stages.

Questions/problems for practice:

Q.10 Explain integer and floating point pipeline stages.

Learning from the lecture 'Integer and Floating Point stages':

Student will be able to understand the need of pipeline architecture to improve the performance of the processor.

Lecture: 36

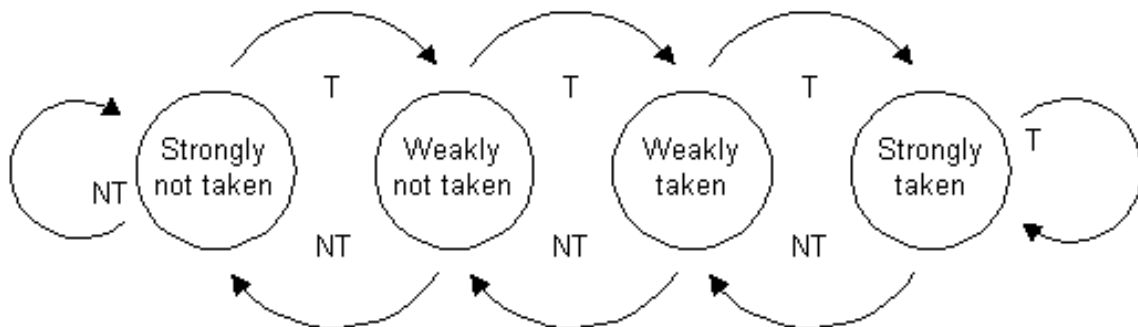
Branch Prediction Logic

Learning Objective: In this lecture students will be able to understand the need of Branch Prediction Logic.

9.6 Dynamic branch prediction problem in Pentium:

- Pentium Processor uses Branch Target Buffer (BTB) to predict the outcome of branch instructions which minimizes pipeline stalls due to prefetch delays
- When a branch is correctly predicted, no performance penalty is incurred.
- But if prediction is not correct, it causes a **3 cycle** penalty in **U pipeline** and **4 cycle** penalty in **V pipeline**.
- When a **call or condition jump** is mispredicted, a **3 clock** penalty is incurred
- BTB is a cache with 256 entries.
- The directory entry for each line contains the following information:
 - A **Valid Bit** that indicates whether or not the entry is in use.
 - **History Bits** that tracks how often the branch is taken
 - **Source memory address** that the branch instruction was fetched from
- BTB sits off to the side of D1 stages of two pipelines and monitors for branch instructions
- The first time a branch instruction enters a pipeline, BTB uses its source address to perform a lookup in cache and this results in BTB miss.
- When instruction reaches the execution stage, the branch will be either taken or not taken.
- If taken, the next instruction should be fetched from the branch target address
- When a branch is taken for first time, the execution unit provides feedback to branch prediction logic and branch target address is recorded in BTB
- A directory entry is made containing source memory address and the history bits.
- The history bit indicates one of the 4 possible states:
 - Strongly Taken
 - Weakly Taken
 - Weakly Not Taken

- Strongly Not Taken
- Strongly Taken:
 - The history bits are initialized to this state when the entry is made first
 - If a branch marked weakly taken is taken again, it is upgraded to strongly taken
 - When a branch marked strongly taken is not taken the next time, it is downgraded to weakly taken
- Weakly Taken
 - If a branch marked weakly taken is taken again, it is upgraded to strongly taken
 - When a branch marked weakly taken is not taken the next time, it is downgraded to weakly not taken
- Weakly Not Taken
 - If a branch marked weakly not taken is taken again, it is upgraded to weakly taken
 - When a branch marked weakly not taken is not taken the next time, it is downgraded to strongly not taken
- Strongly Not Taken
 - If a branch marked strongly not taken is taken again, it is upgraded to weakly not taken
 - When a branch marked strongly not taken is not taken the next time, it remains in strongly not taken state



- During D1 stage of decode, if branch prediction is not taken, no action is taken at this point.

- If prediction taken, the BTB supplies branch target address back to the prefetcher and indicates a positive prediction is being made. In response, prefetcher switches to opposite prefetch queue and immediately begins to prefetch from branch target address
- During execution stage, the branch will either be taken or not.
- The results of the branch are fed back to the BTB and history bits are upgraded or downgraded accordingly.

Let's check the take away from this lecture

9. What is size of Branch Target Buffer?

- | | |
|----------------|-----------------------|
| i) 256 Bytes | ii) 16Bytes |
| iii) 512 Bytes | iv) None of the above |

10. How many levels for branch prediction logic?

- | | |
|--------|-------|
| i) 1 | ii) 3 |
| iii) 2 | iv) 4 |

L24 Exercise:

Q.12 Define pipeline.

Q.13 Write a short note on: BPL.

Question/ problems for practice

Q.14 Explain Dynamic branch prediction logic.

Q.16 How the flushing of instruction queue problem is solved in Pentium 1?

Learning from the Lecture 'BPL':

Student will able to the need and technique of BPL.

Lecture: 37

Comparison of Pentium 2, Pentium 3 and Pentium 4 Processors & Multi core Processors i3, i5 and i7

Learning Objective: In this lecture students will able to compare Pentium family processors and multicore processors.

9.8 Pentium Family:

	<i>Pentium Processor</i>	<i>Pentium Pro Processor</i>	<i>Pentium II Processor</i>	<i>Pentium III Processor</i>	<i>Pentium 4 Processor</i>
Introduced	03/23/93	11/01/95	05/07/97	02/26/99	11/20/00
Operations Per Clock Cycle	2	3	3	5	6
Max Clock Speed	60MHz system bus: 150MHz	60MHz system bus: 180MHz	66MHz system bus: 333MHz	100MHz system bus: 1.0GHz	400MHz system bus: 2.40GHz
Bus Frequency	60MHz, 66MHz	60MHz, 66MHz	66MHz, 100MHz	100MHz, 133MHz	400MHz (100 * 4), 533MHz (133 * 4)
Number of Transistors	3,100,000 (0.8 micron)	5,500,000 (0.35 micron)	7,500,000 (0.35 micron)	24,000,000 (0.13 micron)	42,000,000 (0.13 micron)
L1 Cache	16KB	16KB	32KB	32KB	12k μ op + 8KB Data

L2 Cache	-	1MB	512KB	512KB	512KB
		(on chip)	(off chip)	(on chip)	(on chip)
Addressable Memory	4GB	64GB	64GB	64GB	64GB
Integer Pipelines	2	2	2	2	4
Floating Point Pipelines	1	1	1	1	2
Brief Description	Superscalar architecture brought 5X the performance of the 33MHz Intel486 DX processor Intel's first true server / workstation chip Dual independent bus, dynamic execution, Intel MMX technology Data Prefetch Logic, Level 2 Advanced Transfer Cache Capable of delivering 4.2GB of data-per-second into and out of the processor				

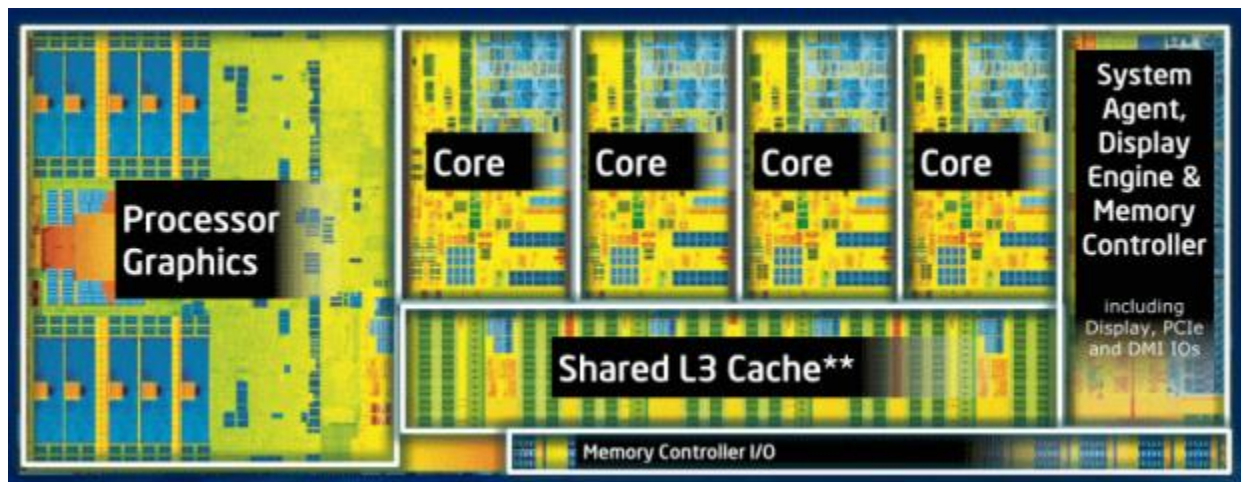
9.9 Multicore Processors: Comparison between i3, i5 and i7 :

Model	Core i3	Core i5	Core i7
Number of cores	2	4	4
Hyper-threading	Yes	No	Yes
Turbo boost	No	Yes	Yes
K model	No	Yes	Yes

Cores

A core can be thought of as an individual processor. A dual-core processor, therefore, has two internal processors, a quad-core model has four. More cores are useful for multi-tasking; for example, you can run two applications at the same time, each one having access to its own dedicated processor.

More cores are also useful for multi-threaded applications, such as video editing. With these types of applications they can use multiple cores to improve performance. Single-threaded applications can only use a single core leaving any others idle. Core i3 processors have two cores, Core i5 CPUs have four and Core i7 models also have four. Some Core i7 Extreme processors have six or eight cores. Generally speaking, we find that most applications can't take full advantage of six or eight cores, so the performance boost from extra cores isn't as great.



Hyper-Threading

Hyper-Threading is Intel's technology for creating two logical cores in each physical core. In other words, to your operating system it appears as though your CPU has double the number of cores than it really does.

In terms of performance, Hyper-Threading speeds up multi-tasking and multi-threaded applications. It's not as fast or as efficient as extra 'real' cores, but it's an improvement over a single Core. Core i3 and i7 processors have this technology, Core i5 processors do not.

Clock Speed

The faster the clock speed in MHz, the faster each core can run. This can create some variances in performance. For example, a Core i3-4370 Haswell processor runs at 3.8GHz. It would be faster running a single-threaded application, which can only use one core, than a Core i5-4590, which only has a clock speed of 3.2GHz. However, running a multi-threaded application, the Core i5 would most likely be quicker, as its four real cores are better than the Core i3's two cores and Hyper-Threading.

Turbo Boost

Turbo Boost is Intel's technology for automatically overclocking a processor, boosting its clock speed higher than the default setting. The CPU monitors its temperature and, when it's running cool enough, will apply the overclock. Core i5 and i7 CPUs have this technology, Core i3 models do not.

K Models

Any CPU that has a model ending with a K means that the CPU is unlocked. This means that you can use BIOS settings to up the clock speed of the chip, overclocking it yourself. We've seen big improvements in performance this way - we pushed the Intel Core i7-4790K chip to 4.7GHz.

Let's check the take away from this lecture

13. How many ALU in Pentium 1.

- i) 1 ii) 2 iii) 3 iv) 4

14. Size of Address bus in P2

- i) 32 bits ii) 36 bits iii) 16 bits iv) none of these

Exercise:

Q.10 Compare Pentium family processors.

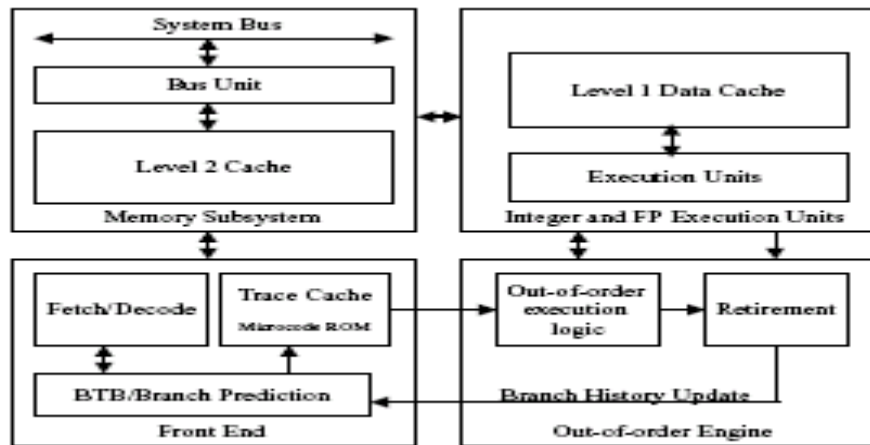
Q.11 Compare Multicore processors.

Learning from the lecture 'Compare Pentium family processors':

Student will be able to differentiate between Pentium family and multicore processors w.r.t. different parameters.

Add to Knowledge (Content Beyond Syllabus)

Intel Net Burst Micro architecture: provides the following important features:



- **Rapid Execution Engine:**
 - Arithmetic Logic Units (ALUs) run at twice the processor frequency.
 - Basic integer operations executes in 1/2 processor clock tick.
 - Provides higher throughput and reduced latency of execution.
- **Hyper Pipelined Technology:**
 - Twenty-stage pipeline to enable industry-leading clock rates for desktop PCs and servers.
 - Provides frequency headroom and scalability to continue leadership into the future.
- **Advanced Dynamic Execution:**
 - Very deep, out-of-order, speculative execution engine.
 - Up to 126 instructions in flight.
 - Up to 48 loads and 24 stores in pipeline.

Enhanced branch prediction capability.

- Reduces the mis-prediction penalty associated with deeper pipelines.
- Advanced branch prediction algorithm.
- 4K-entry branch target array.

New cache subsystem:

First-level caches.

- Advanced Execution Trace Cache stores decoded instructions.
- Execution Trace Cache removes decoder latency from main execution loops.
- Execution Trace Cache integrates path of program execution flow into a single line.
- Low latency data cache with 2 cycle latency.

Second level cache.

- Full-speed, unified 8-way 2nd-Level on-die Advance Transfer Cache.

- Bandwidth and performance increases with processor frequency.

High-performance, quad-pumped bus interfaces to the Intel Net Burst micro architecture system bus.

Support quad-pumped, scalable bus clock to achieve 4X effective speed.

Capable of delivering up to 3.2 GB of bandwidth per second for Pentium(R) 4 processor.

Superscalar issue to enable parallelism. Expanded hardware registers with renaming to avoid register name space limitations. 128-byte cache line size.

5.10 Learning Outcomes:

1. Know:

- Student should be able to define superscalar architecture and superpipelining.
- List the features of Pentium 1, P2, P3 and P4.

2. Comprehend:

- Student should be able to Describe and compare different features of P1,p2,p3 and p4.
- Explain cache organization.

3. Apply, analyze and synthesize:

Student should be able to identify the need for branch prediction logic and use dynamic BPL.

5.11. MCQs

- Because of Pentium's superscalar architecture, the number of instructions that are executed per clock cycle is
a) 1 b) 2 c) 3 d) 4
- The type of execution which means that the CPU should speculate which of the next instructions can be executed earlier is
a) speculative execution
b) out of turn execution
c) dual independent bus
d) multiple branch prediction
- The execution in which the consecutive instruction execution in a sequential flow is hampered is
a) speculative execution
b) out of turn execution
c) dual independent bus
d) multiple branch prediction
- A dual independent bus has
a) Enhanced system bandwidth
b) CPU that can access both cache and memory simultaneously

- c) High throughput
 - d) All of the mentioned
5. The unit that is used to implement the multiple branch prediction in Pentium-Pro is
- a) control unit
 - b) bus interface unit
 - c) branch target buffer
 - d) branch instruction register
6. The unit that accepts the sequence of instructions from the instruction cache as input is
- a) fetch-decode unit
 - b) dispatch-execute unit
 - c) retire unit
 - d) none
7. The instructions that pass through the fetch, decode and execution stages sequentially is known as
- a) sequential instruction
 - b) sequence of fetch, decode and execution
 - c) linear instruction sequencing
 - d) non-linear instruction sequencing
8. Size of data bus in Pentium Processor?
- a) 16 bits b) 32 bits c) 64 bits d) 48 bits
9. How much physical memory can be accessed by Pentium 2 processor?
- (a) 64 MB (b) 64 GB (c) 4 GB (d) 64 TB
10. How much virtual memory can be accessed by Pentium processor?
- (a) 64 MB (b) 64 GB (c) 4 GB (d) 64 TB
11. Size of cache memory in Pentium 1?
- a) 8 KB b) 16 KB c) 32 KB d) 4 GB
12. Size of address bus in Pentium IV ?
- a) 16 bits b) 32 bits c) 64 bits d) 36 bits
13. How many Integer pipeline stages supported by Pentium 1?
- a) 6 b) 5 c) 7 d) 4
14. What size of instruction queue in Pentium I processor?
- a) 256 Bytes b) 128 Bytes c) 16 Bytes d) None of the above

15. How many ALUs are present in Pentium 1 processor?
- a) 2 b)1 c)4 d) 3

5.12 Short Answer questions

- Q.1 what are features of Pentium 1?
Q.2 List Integer pipeline stages of P1.
Q.3 List floating point pipeline stages.
Q.4 List Pentium family processors.
Q.5 Differnciate between Von-numenn and Data flow architecture?

5.13. Long Answer question

- Q.1 Draw and explain the architecture of Pentium 1.
Q.2 Explain integer and floating point pipeline stages of Pentium 1.
Q.3 Compare Pentium family processors.
Q.4 Compare Multicore processors.
Q.5 Explain Superscalar architecture of Pentium Processor.

5.15. References

Pentium Processor by James Antonokos.

Practice for Module No.4 Pentium Processor (based on University Patterns)

- Q.1 A) Draw block diagram of Pentium1. (10marks)
Q.2 A) Explain integer pipeline stages of Pentium 1. (5 marks)
B) Explain floating point pipeline stages of Pentium 1. (5 marks)
Q.3 A) How the flushing of Pipeline problem solved in Pentium1? (10 marks)
Q.4 A) Explain code and data cache organization. (10 marks)
Q.5 A) Compare Intel family processors . (10 marks)
B) Compare Multicore processors. (5 marks).

Self-Assessment

- Q.1 Explain superscalar and super pipeline architecture.
Q.2 List the features of Pentium 4.
Q.3 Compare i3,i5 and i7 processors.
Q.4 Compare Pentium 1 and Pentium 4.

Self-Evaluation

Name of Student		
Class		
Roll No.		
Subject		
Module No.		
S.No		Tick Your choice

22.	Do you understand the term superscalar architecture ?	<input type="radio"/> Yes <input type="radio"/> No
23.	Do you understand the concept pipelining?	<input type="radio"/> Yes <input type="radio"/> No
24.	Do you understand the cache subsystem?	<input type="radio"/> Yes <input type="radio"/> No
25.	Do you able to compare Pentium family processors?	<input type="radio"/> Yes <input type="radio"/> No
26.	Do you able to compare multicore processore?	<input type="radio"/> Yes <input type="radio"/> No

Module: 6

The Microcontroller 8051

6.1. Motivation:

At the completion of the subject, students should be able to:

At the completion of this module, students should be able to:

- Differentiate between Microprocessor and Microcontroller.
- Compare RISC and CISC architecture.
- Distinguish between Von-Neumann and Harvard architecture.

6.2. Syllabus:

Lecture	Content	Duration	Self Study
38 &39	Introduction to 8051 Microcontroller	1 Lecture	2 hours
40	Architecture	2 Lecture	2 hours
41	Pin configuration	1 Lecture	2 hours
42	Memory organization	1 Lecture	2 hours
43	Input /Output Ports	1 Lecture	2 hours
44	Serial communication	1 Lecture	2 hours
45	Interrupts	1 Lecture	2 hours

6.3. Weightage: 20 Marks

6.4. Learning Objectives: Students should be able to-

1. List the different features of 8051 controller.(R)
2. Distinguish between Microprocessor and Microcontroller.(U)
3. Explain the software and hardware architecture of 8051. (E)
- 4.Explain Memory organization details of 8051.(E)

5. Describe the Pin diagram and various functionality of 8051. (U)
6. Use of interrupt in 8051. (A)

6.5. Theoretical Background:

COMPUTER Programms:

A set of instructions written in a specific sequence for the computer to solve a specific task is called a program and software is a collection of such programs. The program stored in the computer memory in the form of binary numbers is called machine instructions. The machine language program is called object code. An assembly language is a mnemonic representation of machine language. Machine language and assembly language are low level languages and are processor specific. The assembly language program the programmer enters is called source code. The source code (assembly language) is translated to object code (machine language) using assembler. Programs can be written in high level languages such as C, C++ etc. High level language will be converted to machine language using compiler or interpreter. Compiler reads the entire program and translate into the object code and then it is executed by the processor. Interpreter takes one statement of the high level language as input and translate it into object code and then executes.

6.6 Abbreviations:

MCS: Microcontroller

RISC : Reduced Instruction Set Computer

CISC : Complex Instruction Set Computer

PSW : Program Status Word

6.7. Formulae : NA

6.8. Key Definitions:

Microcontroller: A microcontroller (abbreviated MCU or μ C) is a computer system on a chip that does a job. It contains an integrated processor, memory (a small amount of RAM, program memory, or both), and programmable input/output peripherals, which are used to interact with things connected to the chip.

6.9. Course Content:

Lecture: 38 , 39 & 40

Introduction to 8051 Microcontroller& Architecture

Learning Objective: In this lecture student will able to learn hardware and software architecture of 80511324601.

5.9.1 Block Diagram

MICROPROCESSORS AND MICROCONTROLLERS

MICROPROCESSORS	MICROCONTROLLERS
Microprocessor contains ALU, General purpose registers, stack pointer, program counter, clock timing circuit, interrupt circuit	Microcontroller contains the circuitry of microprocessor, and in addition it has built in ROM, RAM, I/O Devices, Timers/Counters etc.
It has many instructions to move data between memory and CPU	It has few instructions to move data between memory and CPU
Few bit handling instruction	It has many bit handling instructions
Less number of pins are multifunctional	More number of pins are multifunctional
Single memory map for data and code (program)	Separate memory map for data and code (program)
Access time for memory and I/O are more	Less access time for built in memory and I/O
Microprocessor based system requires additional hardware	It requires less additional hardwares
More flexible in the design point of view	Less flexible since the additional circuits which is residing inside the microcontroller is fixed for a particular microcontroller
Large number of instructions with flexible addressing modes	Limited number of instructions with few addressing modes

RISC AND CISC CPU ARCHITECTURES

Microcontrollers with small instruction set are called reduced instruction set computer (RISC) machines and those with complex instruction set are called complex instruction set computer (CISC). Intel 8051 is an example of CISC machine whereas microchip PIC 18F87X is an example of RISC machine.

RISC	CISC
Instruction takes one or two cycles	Instruction takes multiple cycles
Only load/store instructions are used to access memory	In additions to load and store instructions, memory access is possible with other instructions also
Instructions executed by hardware	Instructions executed by the micro program
Fixed format instruction	Variable format instructions
Few addressing modes	Many addressing mod
Few instructions	Complex instruction set
Most of the have multiple register banks	Single register bank
Highly pipelined	Less pipelined
Complexity is in the compiler	Complexity in the microprogram

HARVARD & VON- NEUMANN CPU ARCHITECTURE

Von-Neumann (Princeton architecture)	Harvard architecture
It uses single memory space for both instructions and data.	It has separate program memory and data memory

It is not possible to fetch instruction code and data	Instruction code and data can be fetched simultaneously
Execution of instruction takes more machine cycle	Execution of instruction takes less machine cycle
Uses CISC architecture	Uses RISC architecture
Instruction pre-fetching is a main feature	Instruction parallelism is a main feature
Also known as control flow or control driven computers	Also known as data flow or data driven computers
Simplifies the chip design because of single memory space	Chip design is complex due to separate memory space
Eg. 8085, 8086	Eg. General purpose microcontrollers, special DSP chips etc.

5.9.2 THE 8051 ARCHITECTURE

Introduction Salient features of 8051 microcontroller are given below.

1. Eight bit CPU
2. On chip clock oscillator
3. 4Kbytes of internal program memory (code memory) [ROM]
4. 128 bytes of internal data memory [RAM]
5. 64 Kbytes of external program memory address space. • 64 Kbytes of external data memory address space.
6. 32 bi directional I/O lines (can be used as four 8 bit ports or 32 individually addressable I/O lines)
7. Two 16 Bit Timer/Counter :T0, T1
8. Full Duplex serial data receiver/transmitter
9. Four Register banks with 8 registers in each bank.

10. Sixteen bit Program counter (PC) and a data pointer (DPTR)
11. 8 Bit Program Status Word (PSW)
12. 8 Bit Stack Pointer
13. Five vector interrupt structure (RESET not considered as an interrupt.)
14. 8051 CPU consists of 8 bit ALU with associated registers like accumulator 'A', B register, PSW, SP, 16 bit program counter, stack pointer.
15. ALU can perform arithmetic and logic functions on 8 bit variables.
16. 8051 has 128 bytes of internal RAM which is divided into
 - Working registers [00 – 1F]
 - Bit addressable memory area [20 – 2F]
 - General purpose memory area (Scratch pad memory) [30-7F]

5.9.3 The 8051 architecture

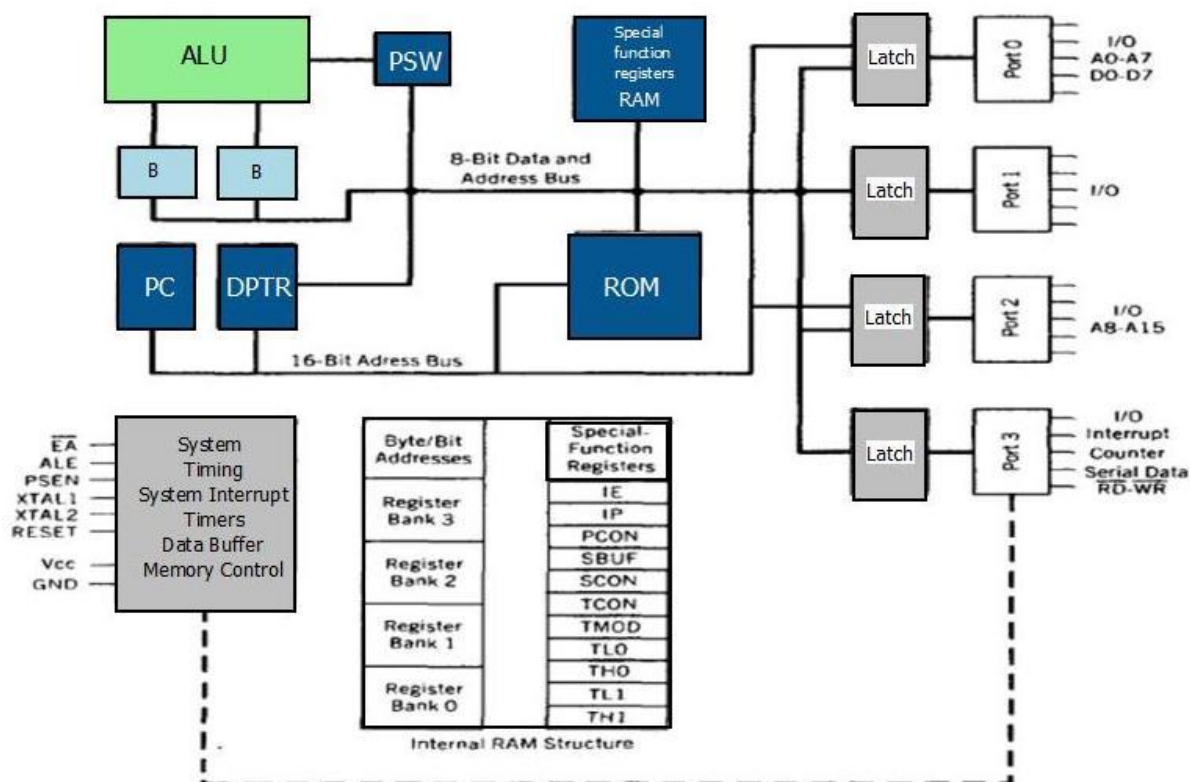


Fig 1 8051 Architecture

Explanation:

1. 8051 has 4 K Bytes of internal ROM. The address space is from 0000 to 0FFFh. If the program size is more than 4 K Bytes 8051 will fetch the code automatically from external memory.
2. Accumulator is an 8 bit register widely used for all arithmetic and logical operations. Accumulator is also used to transfer data between external memory. B register is used along with Accumulator for multiplication and division. A and B registers together is also called MATH registers
3. PSW (Program Status Word): This is an 8 bit register which contains the arithmetic status of ALU and the bank select bits of register banks.

CY AC F0 RS1 RS0 OV - P

CY - carry flag

AC - auxiliary carry flag

F0 - available to the user for general purpose

RS1,RS0 - register bank select bits

OV – overflow

P – parity

4. Stack Pointer (SP) – it contains the address of the data item on the top of the stack. Stack may reside anywhere on the internal RAM. On reset, SP is initialized to 07 so that the default stack will start from address 08 onwards.
5. Data Pointer (DPTR) – DPH (Data pointer higher byte), DPL (Data pointer lower byte). This is a 16 bit register which is used to furnish address information for internal and external program memory and for external data memory.
6. Program Counter (PC) – 16 bit PC contains the address of next instruction to be executed. On reset PC will set to 0000. After fetching every instruction PC will increment by one.

Brief description of Registers:

The 8085 includes six registers, one accumulator and one flag register, as shown in Fig. 3. In addition, it has two 16-bit registers: stack pointer and program counter. They are briefly described

as follows. The 8085 has six general-purpose registers to store 8-bit data; these are identified as B, C, D, E, H and L. they can be combined as register pairs - BC, DE and HL to perform some 16-bit operations. The programmer can use these registers to store or copy data into the register by using data copy instructions.

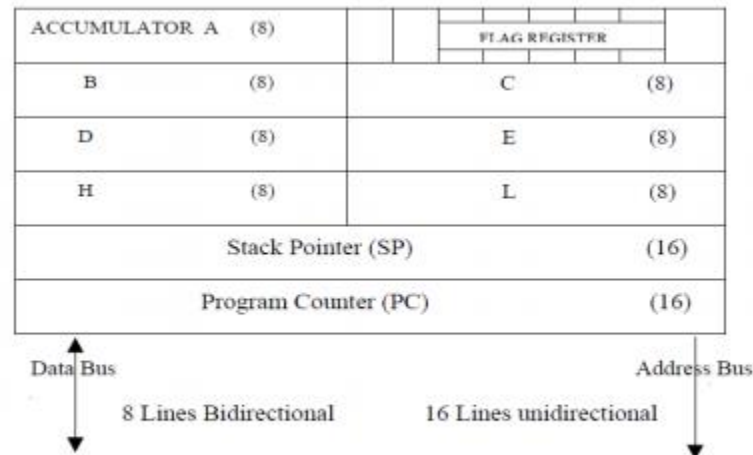


Fig. 3 Register organisation

Let's check the take away from this lecture

- 1) Size of data bus in 8051 ?
 - i) 32 bits
 - ii) 8 bits
 - iii) 16 bits
 - iv) none of the above
- 2) Size of Internal ROM in 8051 Microcontroller?
 - i) 2 K
 - ii) 16K
 - iii) 8K
 - iv) 4K

L21. Exercise:

Q.1 List the features of 8051 Microcontroller.

Q.2 Explain register set of 8051.

Questions/problems for practice:

Q. 3 Differentiate between Microprocessor and Microcontroller.

Learning from the lecture 'Software and Hardware Architecture':

Student will able to explain the hardware architecture of 8051 Microcontroller.

Lecture: 40

8051 Microcontroller: Pin Diagram

Pin Diagram of 8051:

Properties:

1. It is a 8-bit microprocessor
2. Manufactured with N-MOS technology
3. 40 pin IC package
4. It has 16-bit address bus and thus has $2^{16} = 64$ KB addressing capability.
5. Operate with 3 MHz single-phase clock
6. +5 V single power supply The logic pin layout and signal groups of the 8085 microprocessor are shown in Fig. 6. All the signals are classified into six groups:
7. Address bus • Data bus • Control & status signals • Power supply and frequency signals • Externally initiated signals • Serial I/O signals

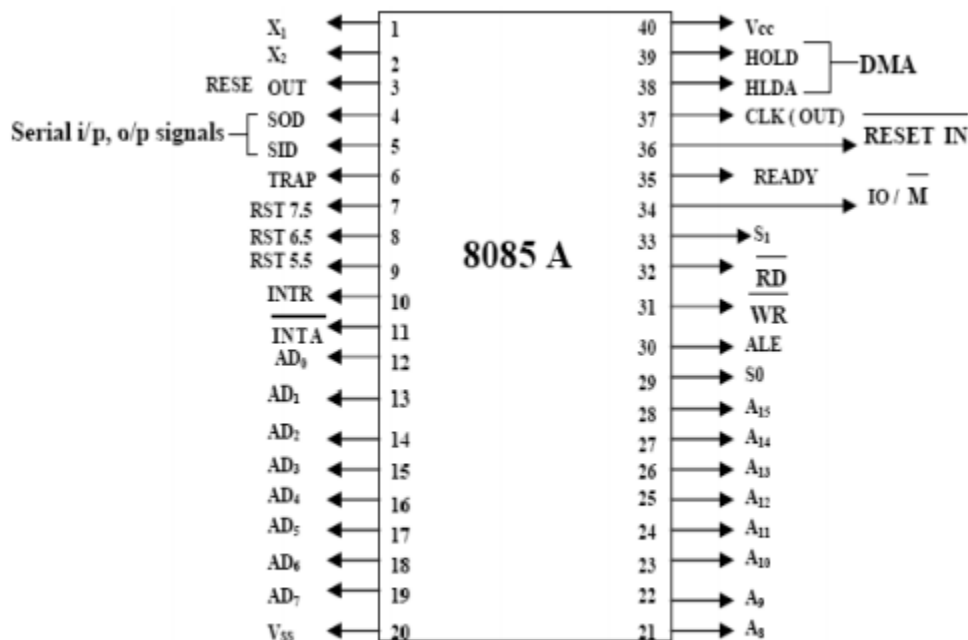


Fig. 6 8085 microprocessor pin layout and signal groups

Functions Description:

1. Address and Data Buses:

- A8 – A15 (output, 3-state): Most significant eight bits of memory addresses and the eight bits of the I/O addresses. These lines enter into tri-state high impedance state during HOLD and HALT modes.
- AD0 – AD7 (input/output, 3-state): Lower significant bits of memory addresses and the eight bits of the I/O addresses during first clock cycle. Behaves as data bus during third and fourth clock cycle. These lines enter into tri-state high impedance state during HOLD and HALT modes.

2. Control & Status Signals:

- ALE: Address latch enable
- RD : Read control signal.
- WR : Write control signal.
- IO/M , S1 and S0 : Status signals. Power Supply & Clock Frequency:
- Vcc: +5 V power supply
- Vss: Ground reference
- X1, X2: A crystal having frequency of 6 MHz is connected at these two pins
- CLK: Clock output Externally Initiated and Interrupt Signals:
- RESET IN : When the signal on this pin is low, the PC is set to 0, the buses are tristated and the processor is reset.
- RESET OUT: This signal indicates that the processor is being reset. The signal can be used to reset other devices.

- **READY:** When this signal is low, the processor waits for an integral number of clock cycles until it goes high.
- **HOLD:** This signal indicates that a peripheral like DMA (direct memory access) controller is requesting the use of address and data bus.
- **HLDA:** This signal acknowledges the HOLD request.
- **INTR:** Interrupt request is a general-purpose interrupt.
- **INTA :** This is used to acknowledge an interrupt.
- **RST 7.5, RST 6.5, RST 5.5 – restart interrupt:** These are vectored interrupts and have highest priority than INTR interrupt.
- **TRAP:** This is a non-maskable interrupt and has the highest priority. Serial I/O Signals:
- **SID:** Serial input signal. Bit on this line is loaded to D7 bit of register A using RIM instruction.
- **SOD:** Serial output signal. Output SOD is set or reset by using SIM instruction.

Let's check the take away from this lecture

- 1) Number of pins in 8051?.
 - i) 32
 - ii) 40
 - iii) 34
 - iv) 50
- 2) Size of address bus and addressable memory in 8051?
 - i) 16 bits, 64 KB
 - ii) 8 bit, 256 Bytes
 - iii) 20 bits, 1 MB
 - iv) None of the above

L21. Exercise:

Q.1 Write functions of various pins of 8051.

Questions/problems for practice:

Q. 3 List various pins of Interrupt handling.

Learning from the lecture 'Pin Diagram':

Student will be able to explain the various pins of 8051..

Lecture: 42

Memory Organization

6.9.1 INTERNAL MEMORY

1. A functioning computer must have memory for program code bytes, commonly in ROM, and RAM memory for variable data that can be altered as the program runs.
2. 8051 has internal RAM (128 bytes) and ROM (4Kbytes)
3. 8051 uses the same address but in different memories for code and data
4. Internal circuitry access the correct memory based on the nature of the operation in progress
5. Can add memory externally if needed

Internal RAM organization

R7	1F
R6	1E
R5	1D
R4	1C
R3	1B
R2	1A
R1	19
R0	18
BANK 3	
R7	17
R6	16
R5	15
R4	14
R3	13
R2	12
R1	11
R0	10
BANK 2	
R7	0F
R6	0E
R5	0D
R4	0C
R3	0B
R2	0A
R1	09
R0	08
BANK 1	
R7	07
R6	06
R5	05
R4	04
R3	03
R2	02
R1	01
R0	00
BANK 0	

Working Registers

Register Banks: 00h to 1Fh.

2F	7F						78
2E	77						70
2D	6F						68
2C	67						60
2B	5F						58
2A	57						50
29	4F						48
28	47						40
27	3F						38
26	37						30
25	2F						28
24	27						20
23	1F						18
22	17						10
21	0F						08
20	07						00

Bit addressable memory

7F
7E
.
.
.
.
.
.
.
.
32
31
30

General purpose memory

The 8051 uses 8 general-purpose registers R0 through R7 (R0, R1, R2, R3, R4, R5, R6, and R7). There are four such register banks. Selection of register bank can be done through RS1, RS0 bits of PSW. On reset, the default Register Bank 0 will be selected. Bit Addressable RAM: 20h to 2Fh
The 8051 supports a special feature which allows access to bit variables. This is where individual memory bits in Internal RAM can be set or cleared. In all there are 128 bits numbered 00h to 7Fh. Being bit variables any one variable can have a value 0 or 1. A bit variable can be set with a command such as SETB and cleared with a command such as CLR.

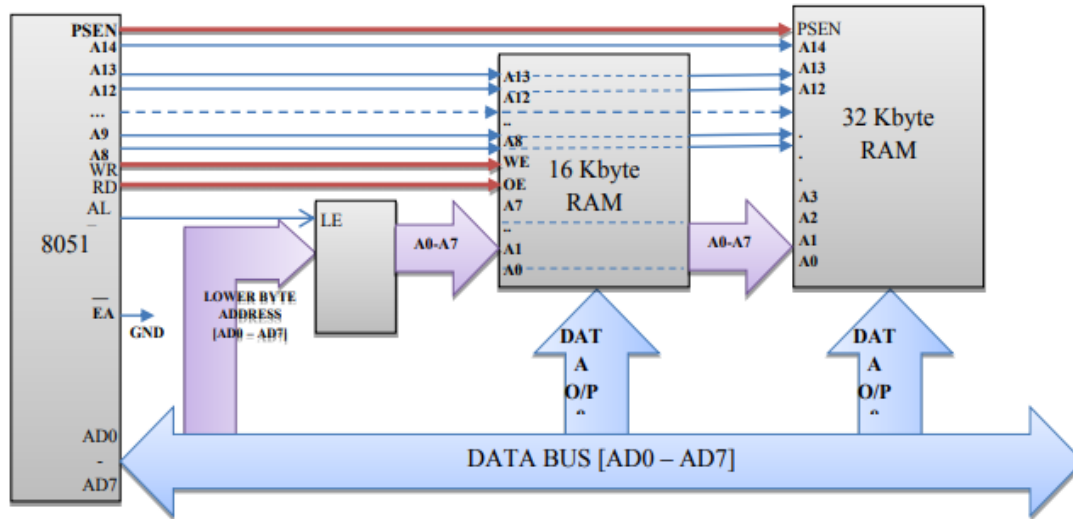
Example instructions are: SETB 25h ; sets the bit 25h (becomes 1) CLR 25h ; clears bit 25h (becomes 0)

The Bit Addressable area of the RAM is just 16 bytes of Internal RAM located between 20h and 2Fh

General Purpose RAM: 30h to 7Fh. Even if 80 bytes of Internal RAM memory are available for general-purpose data storage, user should take care while using the memory location from 00 -2Fh since these locations are also the default register space, stack space, and bit addressable space. It is a good practice to use general purpose memory from 30 – 7Fh. The general purpose RAM can be accessed using direct or indirect addressing modes.

6.9.2 EXTERNAL MEMORY INTERFACING

Eg. Interfacing of 16 K Byte of RAM and 32 K Byte of EPROM to 8051 Number of address lines required for 16 Kbyte memory is 14 lines and that of 32Kbytes of memory is 15 lines. The connections of external memory is shown below.



The lower order address and data bus are multiplexed. De-multiplexing is done by the latch. Initially the address will appear in the bus and this latched at the output of latch using ALE signal. The output of the latch is directly connected to the lower byte address lines of the memory. Later data will be available in this bus. Still the latch output is address it self.

The higher byte of address bus is directly connected to the memory. The number of lines connected depends on the memory size.

The RD and WR (both active low) signals are connected to RAM for reading and writing the data.

PSEN of microcontroller is connected to the output enable of the ROM to read the data from the memory.

EA (active low) pin is always grounded if we use only external memory. Otherwise, once the program size exceeds internal memory the microcontroller will automatically switch to external memory

TIMERS AND COUNTERS:

Timers/Counters are used generally for

- Time reference
- Creating delay
- Wave form properties measurement
- Periodic interrupt generation

- Waveform generation 8051 has two timers, Timer 0 and Timer 1.

Let's check the take away from this lecture

- 1) On power up, the 8051 uses which RAM locations for register R0- R7
 - a) 00-2F
 - b) 00-07
 - c) 00-7F
 - d) 00-0F

- 2) How many bytes of bit addressable memory is present in 8051 based microcontrollers?
 - a) 8 bytes
 - b) 32 bytes
 - c) 16 bytes
 - d) 128 bytes

L21. Exercise:

Q.1 Explain Internal memory organization in 8051.

Questions/problems for practice:

Q. 2 Draw Interfacing of external memory with 8051.

Learning from the lecture 'Memory Organization':

Student will able to explain the internal and external memory organization.

Lecture: 43

Input /Output Ports

Learning Objective: In this lecture student will able to learn I/O Ports.

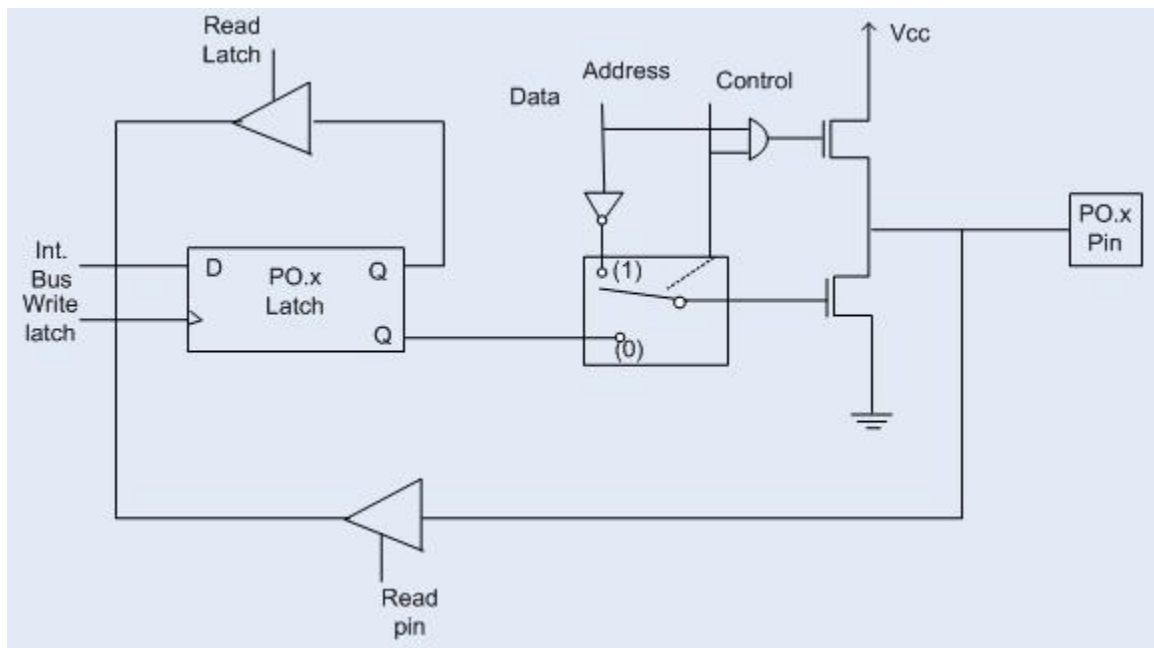
6.9.1 The 8051 has four important ports. Port 0, Port 1, Port 2 and Port 3. These ports allow the microcontroller to connect with the outside world. The four ports of 8051 microcontrollers have certain specific functions and corresponding features.

Features:

- Each port has 8 pins. Thus the four ports jointly comprise 32 pins.
- All ports are bidirectional.
- They are constructed with a D type output latch. They have output drivers and input buffers.
- We can modify their functions using software and hardware that they connect to.
- All the ports are configured as input ports on Reset.
- To configure ports as an input port 1 must be written to that port
- To configure it as an output port 0 must be written to it.

A. PORT 0 :

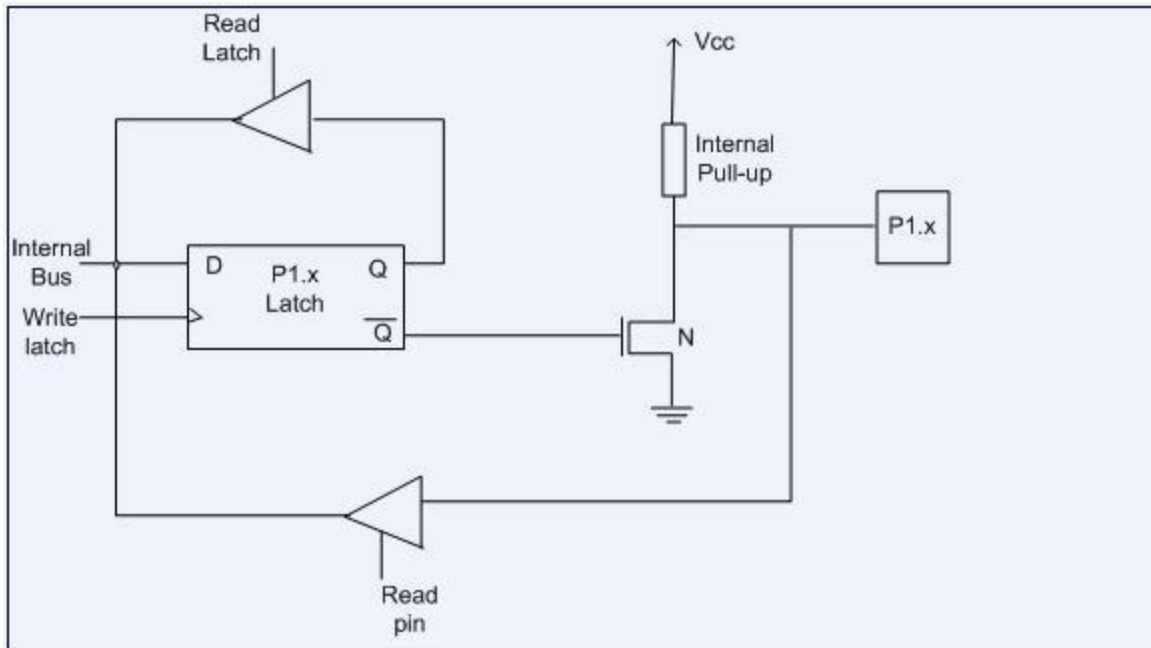
The structure of a Port-0 pin is shown in fig 6. It has 8 pins (P0.0-P0.7).



Port-0 can be used as a normal bidirectional I/O port or it can be used for address/data interfacing for accessing external memory. When control is '1', the port is used for address/data interfacing. When the control is '0', the port can be used as a bidirectional I/O port.

B. PORT 1:

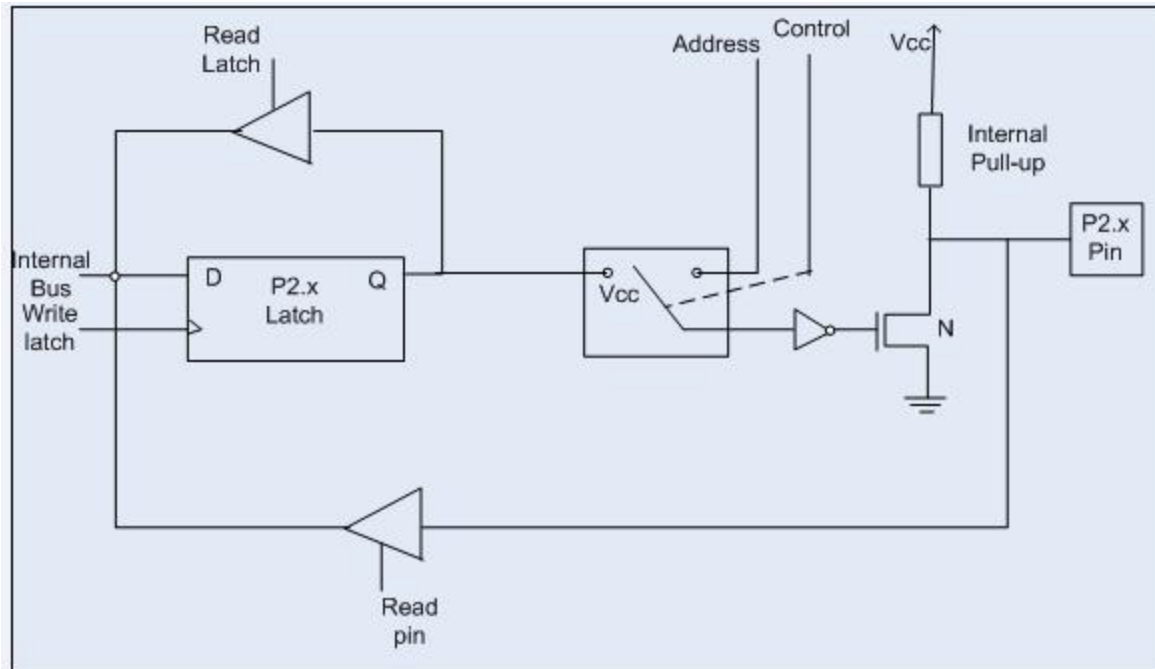
The structure of a port-1 pin is shown in fig below. It has 8 pins (P1.1-P1.7) .



Port-1 dedicated only for I/O interfacing. When used as output port, not needed to connect additional pull-up resistor like port 0. It have provided internally pull-up resistor as shown in fig. below. The pin is pulled up or down through internal pull-up when we want to initialize as an output port. To use port-1 as input port, '1' has to be written to the latch. In this input mode when '1' is written to the pin by the external device then it read fine. But when '0' is written to the pin by the external device then the external source must sink current due to internal pull-up. If the external device is not able to sink the current the pin voltage may rise, leading to a possible wrong reading.

C. PORT 2:

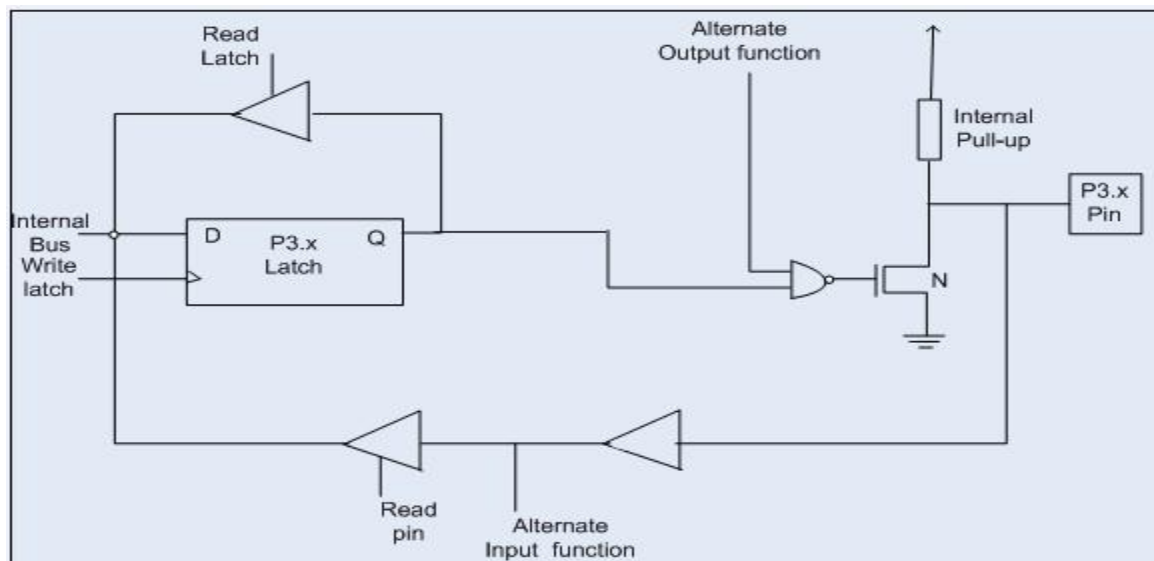
The structure of a port-2 pin is shown in fig. below. It has 8-pins (P2.0-P2.7) .



Port-2 we use for higher external address byte or a normal input/output port. The I/O operation is similar to Port-1. Port-2 latch remains stable when Port-2 pin are used for external memory access. Here again due to internal pull-up there is limited current driving capability.

D. PORT 3:

Port-3 (P3.0-P3.7) having alternate functions to each pin, The internal structure of a port-3 pin is shown in fig below.



It work as an IO port same like Port 2. only alternate function of port 3 makes its architecture different than other ports.

Let's check the take away from this lecture

1) The 8051 microcontroller has ---parallel I/O ports?

a)2 b)3 c)4 d)5

L21. Exercise:

Q.1 What is the use of I/O ports?

Questions/problems for practice:

Q. 2 Explain I/O ports with example.

Learning from the lecture 'I/O ports':

Student will able to explain the I/O ports in 8051.

Lecture: 44

Serial Communication

Learning Objective: In this lecture student will able to learn 8051 serial communication.

6.9.1 Serial Communication

Data Communication:

The 8051 microcontroller is parallel device that transfers eight bits of data simultaneously over eight data lines to parallel I/O devices. Parallel data transfer over a long is very expensive. Hence, a serial communication is widely used in long distance communication. In serial data communication, 8-bit data is converted to serial bits using a parallel in serial out shift register and then it is transmitted over a single data line. The data byte is always transmitted with least significant bit first.

BASICS OF SERIAL DATA COMMUNICATION

Communication Links

1. Simplex communication link: In simplex transmission, the line is dedicated for transmission. The transmitter sends and the receiver receives the data.
2. Half duplex communication link: In half duplex, the communication link can be used for either transmission or reception. Data is transmitted in only one direction at a time.
3. Full duplex communication link: If the data is transmitted in both ways at the same time, it is a full duplex i.e. transmission and reception can proceed simultaneously. This communication link requires two wires for data, one for transmission and one for reception.

Types of Serial communication:

Serial data communication uses two types of communication.

1. Synchronous serial data communication: In this transmitter and receiver are synchronized. It uses a common clock to synchronize the receiver and the transmitter. First the synch character is sent and then the data is transmitted. This format is generally used for high speed transmission. In Synchronous serial data communication a block of data is transmitted at a time.
2. Asynchronous Serial data transmission: In this, different clock sources are used for transmitter and receiver. In this mode, data is transmitted with start and stop bits. A transmission begins with start bit, followed by data and then stop bit. For error checking purpose parity bit is included just prior to stop bit. In Asynchronous serial data communication a single byte is transmitted at a time.
3. Baud rate: The rate at which the data is transmitted is called baud or transfer rate. The baud rate is the reciprocal of the time to send one bit. In asynchronous transmission, baud rate is not equal to number of bits per second. This is because; each byte is preceded by a start bit and followed by parity and stop bit. For example, in synchronous transmission, if data is transmitted with 9600 baud, it means that 9600 bits are transmitted in one second. For bit transmission time = $1 \text{ second} / 9600 = 0.104 \text{ ms}$.

6.9.2 8051 Serial Communication

The 8051 supports a full duplex serial port.

Three special function registers support serial communication.

1. SBUF Register: Serial Buffer (SBUF) register is an 8-bit register. It has separate SBUF registers for data transmission and for data reception. For a byte of data to be transferred via the TXD line, it must be placed in SBUF register. Similarly, SBUF holds the 8-bit data received by the RXD pin and read to accept the received data.

2. SCON register: The contents of the Serial Control (SCON) register are shown below. This register contains mode selection bits, serial port interrupt bit (TI and RI) and also the ninth data bit for transmission and reception (TB8 and RB8).

PCON register: The SMOD bit (bit 7) of PCON register controls the baud rate in asynchronous mode transmission.

Let's check the take away from this lecture

1) How many timers/counters 8051 16 bit microcontroller have?

a) 1 b)2 c) 5 d)8

2) Number of ALU's

i) 2

ii) 1

iii) 3

iv) 4

L21. Exercise:

Q.1 List different types of serial communication.

Questions/problems for practice:

Q. 2 Explain 8051 serial communication.

Learning from the lecture 'Serial Communication':

Student will able to explain the serial communication in 8051.

Lecture: 45

Interrupts

Learning Objective: In this lecture student will able to learn Interrupt handling in Interrupt.

6.9.1

The most powerful and important features are interrupts in 8051 microcontroller. In most of the real-time processes, to handle certain conditions properly, the actual task must be halt for some time – it takes required action – and then must return to the main task. For executing such type of programs, interrupts are necessary. It entirely differs from the polling method wherein the processor must check sequentially each device and ask whether the service is required or not while consuming more processor time.

Types of Interrupts in 8051 Microcontroller

The 8051 microcontroller can recognize five different events that cause the main program to interrupt from the normal execution. These five sources of interrupts in 8051 are:

1. Timer 0 overflow interrupt- TF0
2. Timer 1 overflow interrupt- TF1
3. External hardware interrupt- INT0
4. External hardware interrupt- INT1
5. Serial communication interrupt- RI/TI

The Timer and Serial interrupts are internally generated by the microcontroller, whereas the external interrupts are generated by additional interfacing devices or switches that are externally connected to the microcontroller. These external interrupts can be edge triggered or level triggered. When an interrupt occurs, the microcontroller executes the interrupt service routine so that memory location corresponds to the interrupt that enables it.

Let's check the take away from this lecture

- 1) In 8051 which interrupt has highest priority?
a) IE1 b) TF0 c) IE0 d) TF1
- 2) Serial port interrupt is generated, if ____ bits are set

a) IE b) RI, IE c) IP, TI d) RI, TI

L21. Exercise:

Q.1 List different types of Interrupts.

Questions/problems for practice:

Q. 2 How the interrupt get handled in 8051?.

Learning from the lecture 'Interrupt':

Student will able to explain the interrupts and its uapplications.

6.10 Learning Outcomes:

1. Know:
 - a) Student should be able to differentiate between Microprocessor and Microcontroller.
 - b) List the features of 8051 MCS..
2. Comprehend:
 - a) Student should be able to describe the architecture of 8051 MCS.
 - b) Explain the serial communication and I/O ports.
3. Apply, analyze and synthesize:

Student should be able to show the demonstration how to interface external memory with 8051.

6.11. MCQs

1. How many bytes of bit addressable memory is present in 8051 based microcontrollers?
 - a) 8 bytes
 - b) 32 bytes
 - c) 16 bytes
 - d) 128 bytes
2. On power up, the 8051 uses which RAM locations for register R0- R7
 - a) 00-2F
 - b) 00-07
 - c) 00-7F
 - d) 00-0F
3. When the microcontroller executes some arithmetic operations, then the flag bits of which register are affected?
 - a) PSW

- b) SP
 - c) DPTR
 - d) PC
4. When 8051 wakes up then 0x00 is loaded to which register?
- a) DPTR
 - b) SP
 - c) PC
 - d) PSW
5. 8051 series has how many 16 bit registers?
- a) 2
 - b) 3
 - c) 1
 - d) 0
6. The 8051 microcontroller is of ____pin package as a _____ processor. a) 30, 1byte
b) 20, 1 byte c) 40, 8 bit d) 40, 8 byte
7. What is the address range of SFR Register bank? a) 00H-77H b) 40H-80H c) 80H-7FH d) 80H-FFH
8. What is the size of internal RAM memory of 8051 microcontroller?
- a) 32 Bytes b) 64 bytes c) 128 bytes d) 256 bytes
9. Match the following:
- 1) TCON i) contains status information
 - 2) SBUF ii) timer / counter control register
 - 3) TMOD iii) idle bit, power down bit
 - 4) PSW iv) serial data buffer for Tx and Rx.
 - 5) PCON v) timer/ counter modes of operation.

- a) 1->ii, 2->iv, 3->v, 4->i, 5->iii.
- b) 1->i, 2->v, 3->iv, 4->iii, 5->ii.
- c) 1->v, 2->iii, 3->ii, 4->iv, 5->i.
- d) 1->iii, 2->ii, 3->i, 4->v, 5->iv.
10. In 8051 which interrupt has highest priority? a)IE1 b)TF0 c)IE0 d)TF1
11. Serial port interrupt is generated, if ____ bits are set a) IE b) RI, IE c) IP, TI d) RI, TI
12. The SP is of ____ wide register. And this may be defined anywhere in the _____. a) 8 byte, on-chip 128 byte RAM. b) 8 bit, on chip 256 byte RAM. c) 16 bit, on-chip 128 byte ROM d) 8 bit, on chip 128 byte RAM
13. After reset, SP register is initialized to address_____. a) 8H b) 9H c) 7H d) 6H
14. How many timers/counters 8051 16 bit microcontroller have?
- a) 1 b)2 c) 5 d)8
15. The 8051 microcontroller has ---parallel I/O ports?
- a)2 b)3 c)4 d)5

6.12 Short Answer questions

Q.1 Differentiate between microprocessor and Microcontroller.

Q.2 List various interrupts of 8051.

Q.3 Draw and explain register set 8051.

Q.4 Describe serial communication of 8051.

Q.5 List the features of I/O ports.

6.13. Long Answer question

Q.1 Interface 8051 to external ROM and RAM and explain how 8051 access them.

Q.2 Explain briefly interrupt structure of 8051 .

Q.3 Explain the Architecture of 8051.What are the blocks in Microcontroller.

Q.4 Explain the Pin Diagram of 8051.

Q.5 Explain in detail I/O ports structure of 8051.

6.15. References

The 8051 Microcontroller: Architecture, Programming, and Applications **by** Kenneth Ayala J.
The 8051 microcontroller and embedded systems by Mazidi Ali, Muhammad Mazidi Gillispie Janice.

6.16 Practice for Module No.6 Intel 8051 Microcontroller

1. Mention the timers of 8051?
2. Mention the bit addresses of ports p0 and p1?
3. How many bit addressable locations are placed in internal RAM of 8051?
4. Mention the SFR registers used in timer operation?
5. Mention the operating modes of 8051?
6. What is the function of C/T bit in TMOD register?
7. Find the timers clock frequency for the crystal frequency of 11.0592MHz?
8. State the function of M1 and M0 bits in TMOD register?
9. What is the function of TF0 bit in TCON register?
10. State the use of T0 pin of 8051?

Self-Assessment

Q.1 Explain the features of 8051.

Q.2 Differentiate between RISC and CISC.

Self-Evaluation

Name of Student		
Class		
Roll No.		
Subject		
Module No.		
S.No		Tick Your choice
27.	Do you understand the working of 8051?	<input type="radio"/> Yes <input type="radio"/> No
28.	Do you understand the difference between microprocessor and microcontroller?	<input type="radio"/> Yes <input type="radio"/> No
29.	Do you able to differentiate between RISC and CISC?	<input type="radio"/> Yes <input type="radio"/> No
30.	Do you understand module ?	<input type="radio"/> Yes, Completely. <input type="radio"/> Partialy. <input type="radio"/> No, Not at all.